AD-A047 997 RAYTHEON CO BEDFORD MASS MISSILE SYSTEMS DIV F/G 17/9
HIGH SPEED MICRO SIGNAL PROCESSOR STUDY.(U)
SEP 77 G N SHAPIRO F33615-76-C-1339
UNCLASSIFIED AFAL-TR-77-52 NL

1 OF 3
ADA047997

AD-A047997

AFAL-TR-77-52

(1)

HIGH SPEED MICRO SIGNAL PROCESSOR STUDY

Raytheon Company
Missile Systems Division
Bedford, Massachusetts 01730

SEPTEMBER 1977

Final Report for period June 1976 - January 1977

Approved for public release; distribution unlimited.

AIR FORCE AVIONICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Ronald A. Belt

RONALD A. BELT
Project Engineer

Robert M. Werner

ROBERT M. WERNER, Acting Chief
Processor Technology Group

FOR THE COMMANDER

Stanley E Wagner

STANLEY E. WAGNER, Chief
Microelectronics Branch
Electronic Technology Division

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFAL/DHE-1, W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFAL-TR-77-52	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* HIGH SPEED MICRO SIGNAL PROCESSOR STUDY		5. TYPE OF REPORT & PERIOD COVERED Final Report 1 Jun 1976 - 31 Jan 1977 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) DR. GERALD N. SHAPIRO		8. CONTRACT OR GRANT NUMBER(s) F33615-76-C-1339
9. PERFORMING ORGANIZATION NAME AND ADDRESS RAYTHEON COMPANY MISSILE SYSTEMS DIVISION BEDFORD, MASS. 01730		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS PE 62204F 6096-05-75
11. CONTROLLING OFFICE NAME AND ADDRESS AIR FORCE AVIONICS LABORATORY WPAFB, OHIO		12. REPORT DATE September 1977 13. NUMBER OF PAGES 202 incl, Appendix
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Micro Signal Processor
Pipeline Processor
Radar Processor

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This study defines a micro signal processor design based on maximum use of off-the-shelf IC functions, supplemented by special personalization of standard arrays. The signal processor elements can be grouped into Control, Memory and Arithmetic type elements. These elements are stackable into a variety of configurations because of the standardization applied to the data transfer timing.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

The processor design has a 24 bit multiword format with 2's complement arithmetic and processing capability of 25 - 40 million adds/sec (where 1 multiply = 3 adds).

PREFACE

We wish to thank the following people at AFAL DHE for their contribution in initiating, supporting and guiding the development of this high speed micro signal processor, namely:

Dr. J. DeCaire, Lt. G. Mallaley, Dr. R. Belt, J. Shanklin, and R. Coker.

We also wish to thank the following Raytheon personnel for their detailed technical contributions, namely:

H. Sobel, H. Klemmer, B. Goldstone, J. Simone, J. Spicer, B. Eiseman, T. Daleiden, H. Peters, and M. Kajen.

Dr. Gerald N. Shapiro
Program Manager

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Cont.)

LIST OF ILLUSTRATIONS (Cont.)

LIST OF TABLES

SECTION I
INTRODUCTION AND SUMMARY

1.1 Overview

The task of signal processing in advanced avionic weapon systems is to abstract megabits-per-second sensor information into a manageable basis for decision making. Reduction of system size, cost and power, and increase of reliability requires exploiting the latest technology. Operating mode flexibility is equally important for extended visibility into the environment, for performance improvement, for faster reaction time, or for lower ECM vulnerability.

This study defines a micro signal processor design based on maximum use of "off-the-shelf" IC functions, supplemented by special personalizations of standard arrays. A top-down analysis was employed and involved the following tasks:

1) A functional analysis of signal processing tasks at the algorithm and processing task level
2) A performance analysis of representative tasks
3) A state-of-the-art review of industry LSI micro processor development
4) A detailed functional definition of the hardware and software aspects of micro signal processor circuit elements
5) A simulation of the approaches defined above
6) A circuit technology review to isolate trends, and
7) A development plan

An outline of the analysis results in presented in Table 1. A simplistic view of the micro signal processor parts is presented in Figure 1. Sections 1 and 2 of this report will delve into the nature of these elements in more detail.

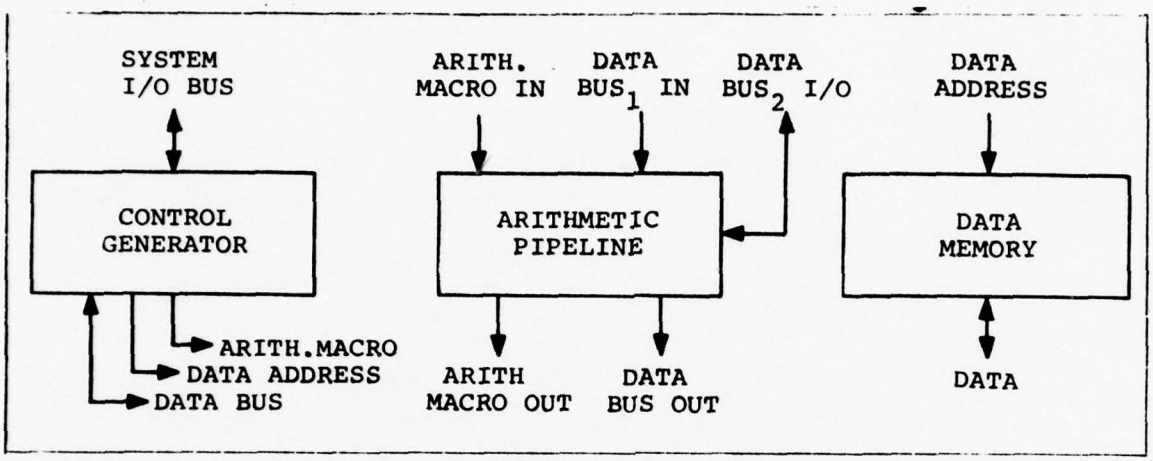


Figure 1 - Micro Signal Processor Parts

A typical application for the μSP will be a minimum configuration with only one of each type element. Such an application arises in the usual system growth shown in Figure 2, from no specialized signal processor, to recognizing the cost-effectiveness of a μSP. First, all signal processing is done by a GP computer. Second, a hardwired FFT box gets introduced. Third, a couple of batch sizes are employed, necessitating a load-while-processing type buffer. From there and a few more system requirement changes a μSP is justifyable.

More powerful signal processing systems are also possible with these same set of building blocks. Figure 3a shows one possible scheme for netting both subsystems and systems. A key concept developed in this study is the signal processor family capability for this design. Family features are summarized in Table 2. The potential thus exists for a compatible Maxi signal processor such a Figure 3b.

TABLE 1
μSP GENERAL DESCRIPTION

• ARCHITECTURE	• Design oriented toward maximum use of bit sliced μP logic
	• Macro Programmable - System Level
	• Micro Programmable - Pipeline Level
	• Pipeline reconfigurable every macro cycle
	• Four clock cycles equal one macro cycle
	• Modular elements of similar size
	• Elements stackable for special configurations
	• Growable to 1 clock = 1 macro
	• Operand routing by scratchpad addressing
	• Simultaneous instruction sequencing, address generation, arithmetic execution and data storage
	• Time shared interconnect
• ARITHMETICS	• 24 bit multiword format
	• Double precision 24 bits
	• Complex arithmetic (12+j12 format)
	• 2's complement arithmetic
	• Block floating point (real/complex)
	• Scaling, Multiply & Dual Adder Elements
	• Double precision complex via multi macro
• THRUPUT	• Processing rate 25-40 MAPS (1 multiply = 3 adds)
	• Clock cycle 150 nsec
	• 1024 pt complex FFT in 3.1 msec (includes all overhead)
• PHYSICAL	• Approx. 200 commercial components
	• Approx. 80 - combination commercial & special LSI
	• Less than 50 watts in commercial/special LSI mix
	• Interface with 16 bit computer

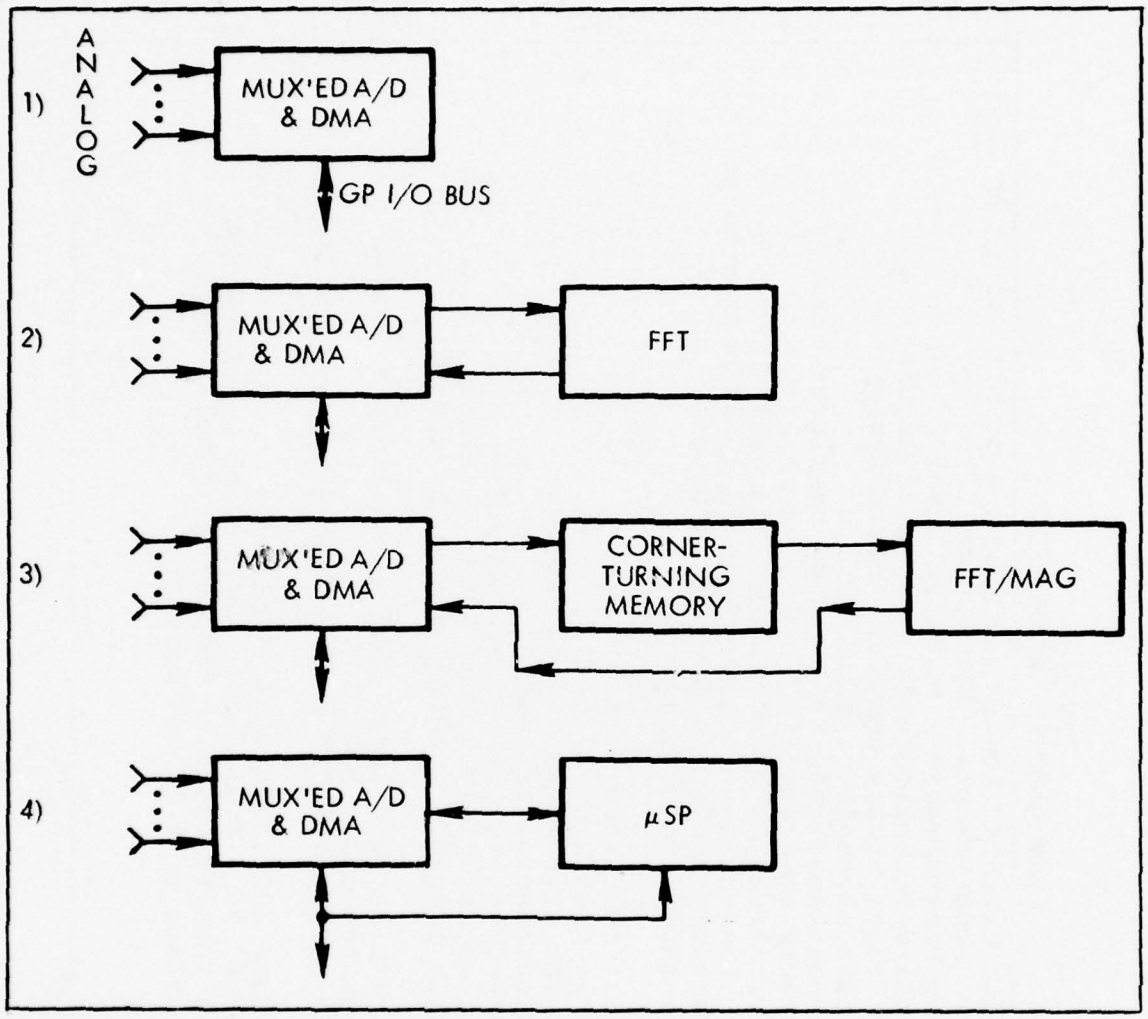


Figure 2 - Small Signal Processor System Evolution

TABLE 2

SIGNAL PROCESSOR FAMILY CONCEPT

- BASIC INSTRUCTION SET PLUS EXPANSION SET
- DATA & I/O FORMAT COMPATIBILITY
- COMMON SUPPORT SOFTWARE
- BUILDING BLOCKS (ELEMENTS) CONFIGURABLE FOR RANGE OF THRUPUTS: MICRO TO MAXI
- PROCESSORS CAN NET IN PARALLEL AND/OR SERIAL
- MEMORY SPEED VS DENSITY EXPLOITABLE
- IMPLEMENTABLE WITH TODAY'S AND TOMORROW'S LSI:
  - COMMERCIAL, ARRAYS, CUSTOM
  - LSTTL, CMOS-SOS, ECL, . . .

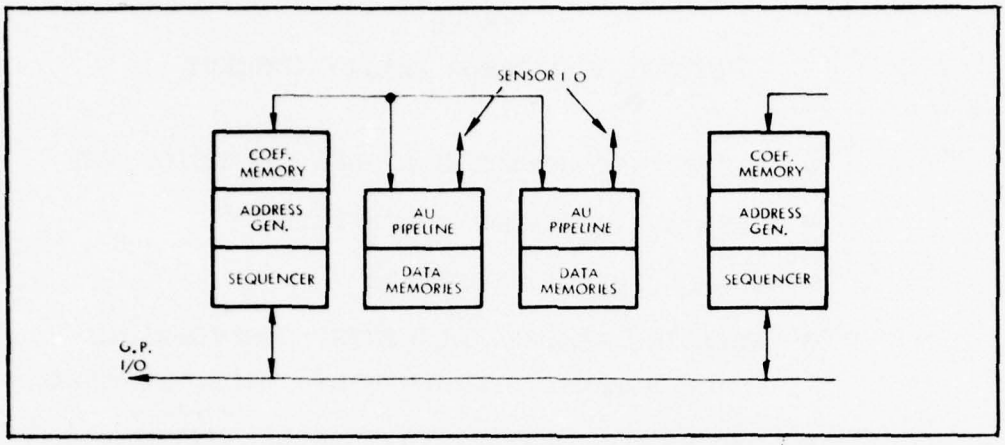


Figure 3a - Netting Subsystems & Systems

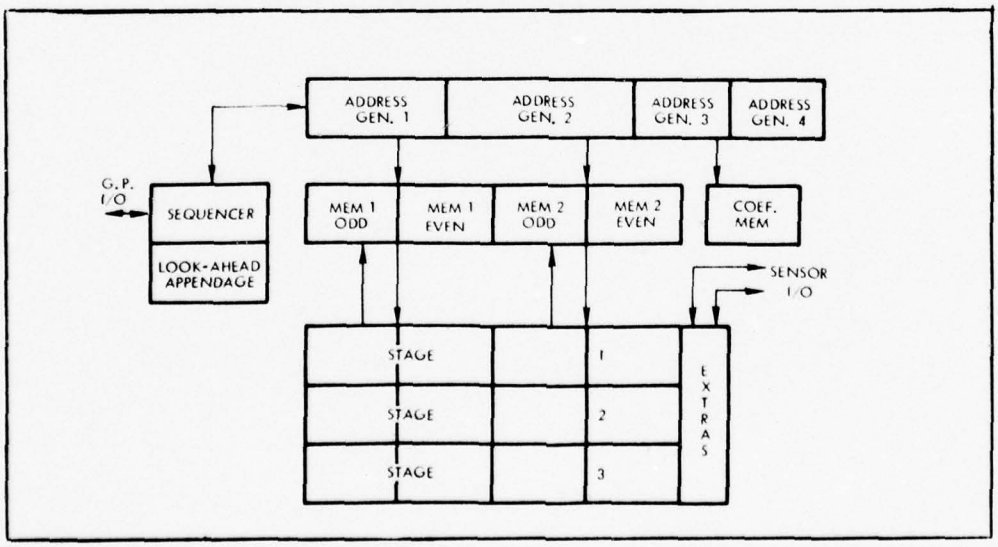


Figure 3b - Maxi Signal Processor

1.2 μSP Characteristics Summary

The μSP elements can be grouped into Control, Memory, and Arithmetic type elements. Table 3 lists the key functions performed by each group of elements. These elements are stackable into a variety of configurations because of the standardization applied to data transfer timing. As shown in Figure 4, the unidirectional data buses transfer data on a four cycle timing basis. These four slots transfer either two complex words or two double length words or one of each. The timing to a memory element or arithmetic element may differ because of the pipelined processing of data, but only by an integer multiple of 4 clocks. Hence, the order of element placement is restricted by what makes computational sense, rather than implementation peculiarity. Such time-sharing of one bus is also very pin efficient, both in inter element connections and in buffer storage pins.

The control is composed of two elements, a sequencer and address generator. The sequencer element takes care of program branching, while the address generator provides both read and write address for the various types of memories. Some of their key features are shown in Table 4. Also shown in Table 4 is a summary of the arithmetic pipeline characteristics. It is composed of three elements, illustrated in Figure 5.

Figure 6 shows the standard interconnection of the system elements, including sequencer, address generator, local memory, bulk memory, coefficient memory and pipeline stages 1, 2, and 3. There is a 16 bit I/O bus which communicates to other μP's or μSP's. There is a pipeline input bus and a pipeline output bus, each of which is 12 bits and transfers data over 4 clock cycles during one macro cycle. The address bus to the coefficient and bulk memories also serves as a bidirectional data bus to connect with the GP I/O, and for load-while-processing operation.

TABLE 3

μSP HARDWARE ELEMENTS

• CONTROL ELEMENTS	• Sets up order of instruction execution & number of times instructions repeated. • 16 bit immediate address fields • Saves selected addresses for sublist processing • Generate read & write addresses for two local memories coefficient memory & bulk memory with proper delays
• MEMORY ELEMENTS	• Local memory has multiport appearance and provides two sources and two destinations for pipeline; and orders memory for time shared interconnects. • Coefficient memory holds constants and function tables; partially writeable from a GP interface at 150 nsec. • Bulk memory accommodates highest density slower speed RAM up to 600 nsec cycle time for efficient processing of large amounts of data.
• ARITHMETIC ELEMENT	• Stage 1 provides scaling based on lead zero manipulation, approximate angle calculation for complex numbers (Single and double precision). • Stage 2 implements data routing and multiply. • Stage 3 has a pair of adders and a shift select to do complex, double precision arithmetic and sorting.

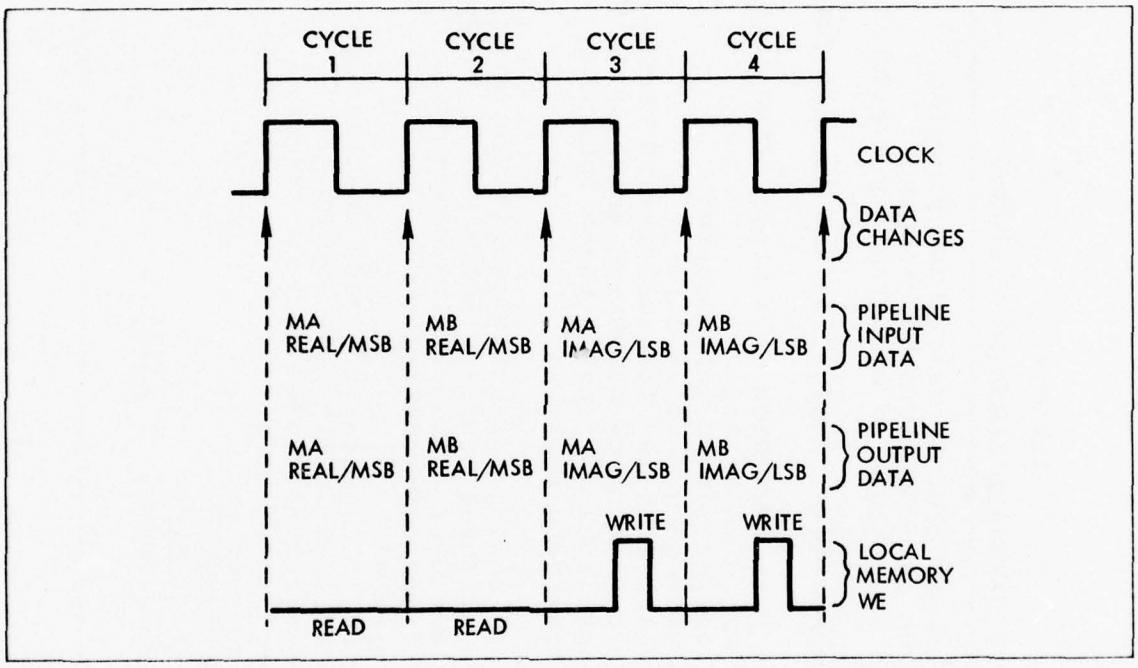


Figure 4 - Four Cycle Timing

TABLE 4

μSP HARDWARE ELEMENTS FEATURES

• SEQUENCER	• Match slow control to multiphase pipeline • Avoid look ahead on loop bookkeeping • Oriented toward use of commercially available bit sliced μP IC's • Upward compatible instruction set for μSP family
• ADDRESS GENERATOR	• Generate four addresses with proper time tags • Independent 4 phase timing • Subroutine parameters accommodated • Local memory growth to 16 bits • Fractional addressing for coefficient memory • Data set selection by buffering selected addresses • Growth potential for single phase operation • GP I/O test/data port
• PIPELINE	• Four clock timing, growth potential to one clock • Composed of three stages, each a useful element • Single length or double length computation • Real or imaginary computations • Fixed point with block scaling • Up to four multiplies and eight adds per macro • Coefficients can enter data stream • Data can enter/influence address stream • Growth to true floating point • Format conversion • Overflow limiting • Normalize numbers • Fast divide by way of multiply

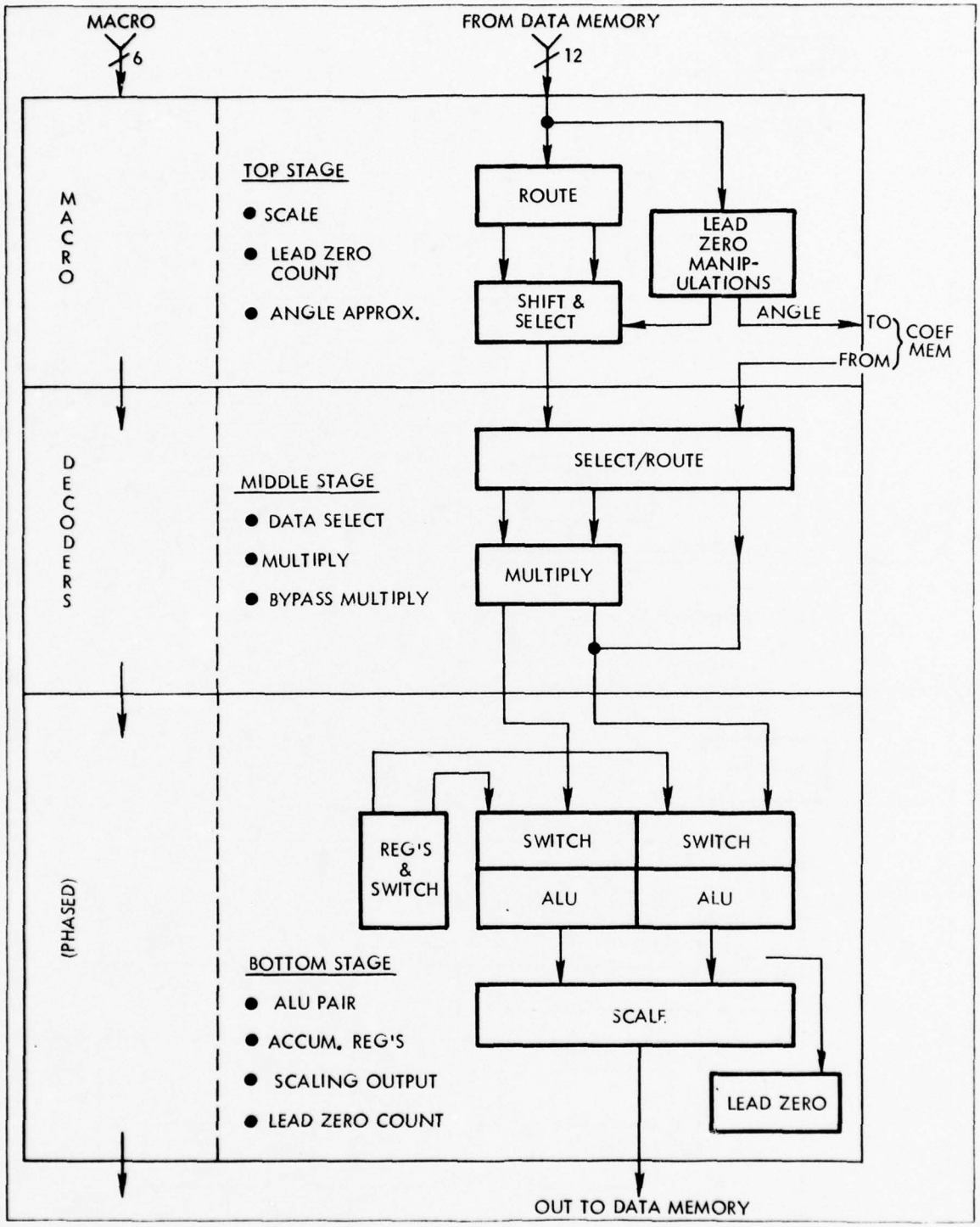


Figure 5 - μSP Arithmetic Pipeline

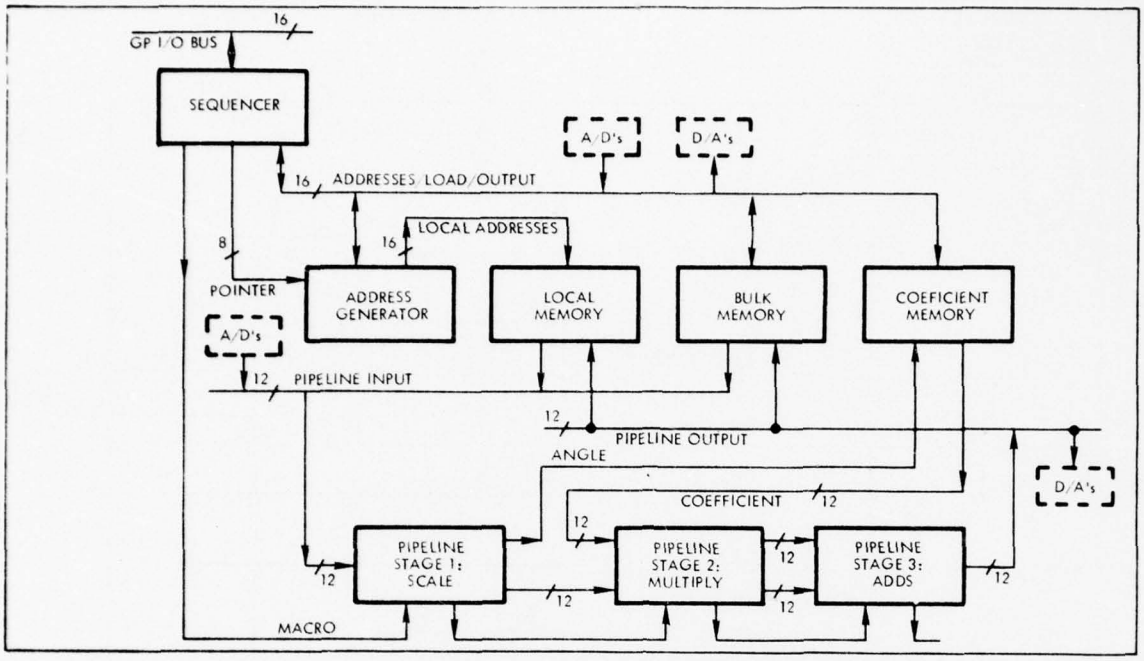


Figure 6 - μSP Full System Interconnect-I

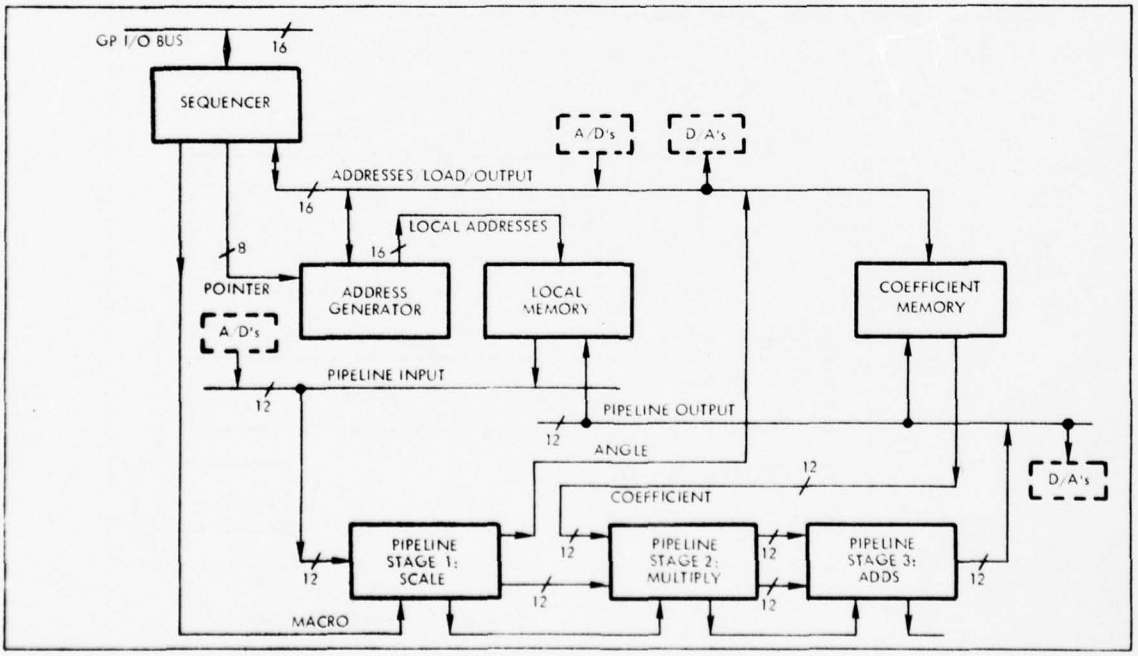


Figure 7 - μSP Full System Interconnect-II

A μSP system can also be configured without a bulk memory element. Figure 7 shows such a configuration, where a signal processor data to GP computer path is possible through the coefficient memory. This alternative provides for the very smallest system configuration, where minimum cost of total logic is important.

Note that the system elements have been defined to minimize the number of pins. For example, two unidirectional buses of 12 bits are used to and from the pipeline rather than one 24 bit bidirectional bus to reduce the number of pins to the pipeline stages. All elements shall fit within a 64 pin interconnection, allowing convenient size hybridization. Further when LSI densities permit, elements may be combined without increasing pin totals. An example is merging pipeline stages 2 and 3 or even stages 1, 2 and 3.

The actual element control has been chosen for straightforwardness rather than minimization of numbers of control bits. Thus the address generator's memory, for example, is a little over 16 bits wide. This simplifies the decoding logic, adds to the intelligibility of the designs, and allows for later optimization with a particular LSI technology implementation. We expect that as more experience is gathered with the μSP design, reductions will suggest themselves. For example, the sequencer element has dropped the operations "DEL" and "PLP" as being too specialized, peculiar to the 8X02 sequencer IC, and unnecessary to typical programs.

A hierachy of software exists when applying the μSP for a given mission. Each mission, such as ground mapping, has several modes. Those modes are composed of common algorithms such as pulse compression or FIR filtering. These algorithms are in turn composed of macro commands such as FFT "butterfly" or pole-pair calculation. The actual micro bits that control adders and multipliers are really invisible to the user. In essence, the macro

commands are the instruction set for this μSP. Table 5 summarizes the capabilities for the μSP firmware/software. Some times associated with various algorithms are presented in Table 6. These times are based on readily available LSTTL implementation using a 150 ns clock. Higher speed technology implementation would be scaled accordingly.

As part of this study, an assembler and simulator effort was undertaken. Effort was concentrated on the control areas, because with the pressure of time, these represented critical areas for validating μSP efficiency. Code was generated in Fortran on the CDC CYBER 73 system. Coding was based on an instruction Set Processor (ISP) description for the sequencer and address generator. More details can be found in the Simulation User's Manual (Raytheon BR-9632), and in chapter 6 of this report. Figure 8 outlines the software simulation flow.

Implementation of these μSP elements was seriously considered. The primary emphasis was on exploiting the available semiconductor industry LSI, both current and visible trends. Recognition of the limitations of relying on the commercial world was also considered, leading to recommended chip types whose designs are quite general and yet significantly reduce the number of total chips. Table 7 lists the major chip types recommended. Table 8 shows the parts count for implementing the μSP with either all commercial IC's today or with the best of available LSI and using Raytheon's 300 gate array IC's to form the recommended chip types. Note that the number of parts per element in each group is about the same, although for different reasons, such as memory bit width limitations or arithmetic complexity limitation.

More detail on the μSP elements is presented in the next chapter, with the backup for its derivation following in Sections III, IV and V.

TABLE 5

μSP FIRMWARE/SOFTWARE

- FIRMWARE
  - 256 macro capability
  - Addressing for 64K words in local memory
  - 1K words of sequencer program memory
  - 1K words of address generator program memory
- MACRO COMMANDS
  - FFT Complex
  - FFT Double Precision (3 step)
  - Time domain weighting
  - Frequency translation
  - Vector manipulations
    - Vector Normalize
    - Vector scale & add
    - Sum of vector elements
    - Vector dot products
    - Peak detect
    - Complex vector multiply
  - Correlation
  - Finite Impulse response filter
  - Infinite impulse response filter
    - 2 zero
    - 2 pole
  - Dual magnitude of complex vectors
  - CFAR
    - Magnitude & sliding window
    - Threshold against largest of adjacent window
- SIMULATOR
  - Instruction set processor (ISP) description for sequencer address generator
  - Fortran coded sequencer & address generator assemblers & simulators

TABLE 6

μSP ALGORITHMS EXAMPLES

	FUNCTION	SIZE	DATA TYPE	TIME*	COMMENT
●	FFT	64	Complex	0.12 msec	-
●	FFT	1024	Complex	3.07 msec	-
	FFT	64	Complex - double precision	0.36 msec	-
●	HETRODYNE & FILTER	N	Complex	3.0N μsec	4 pole & 4 zero
●	DUAL MAGNITUDE	N	Complex	0.3N μsec	< ±2% accuracy
●	CFAR	N	Complex	1.2N μsec	K point window K< N

* Based on 150 nsec clock cycle.

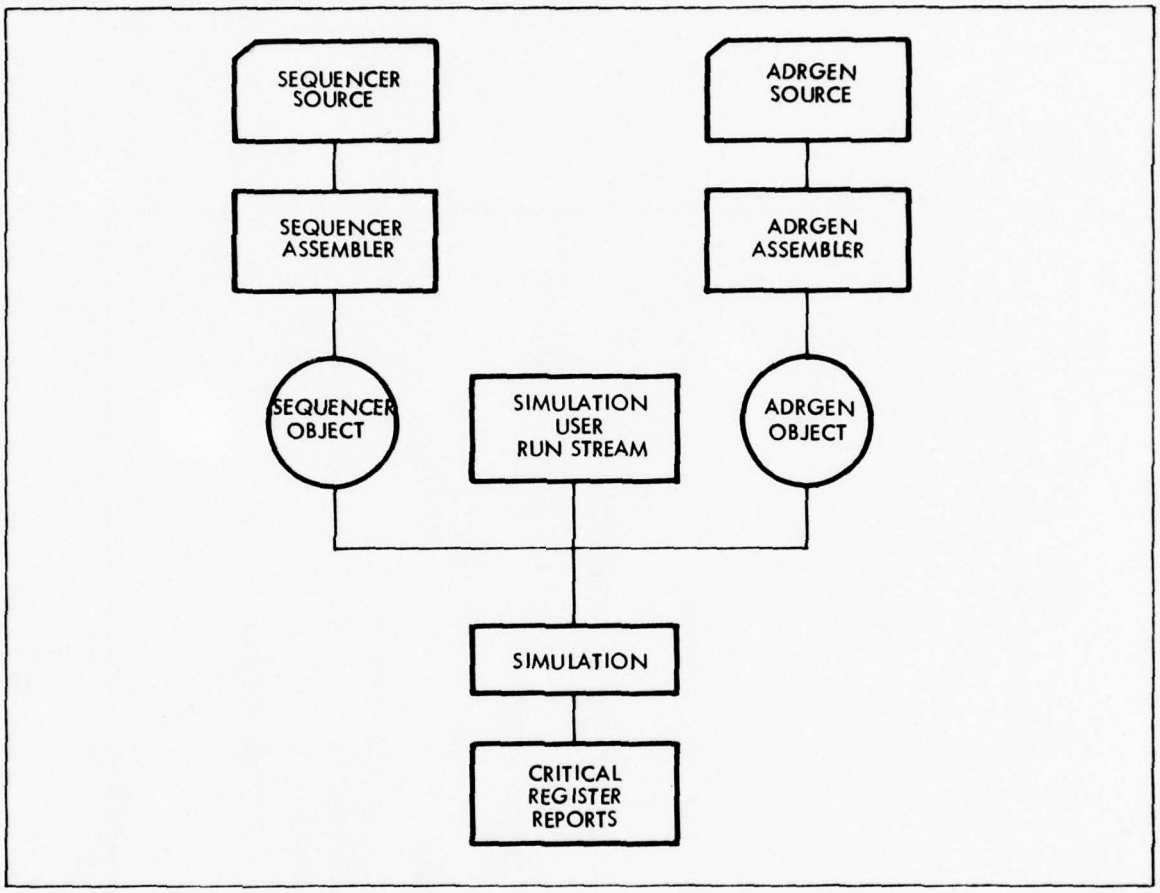


Figure 8 - Simulation Flow

TABLE 7

PROPOSED μSP CHIP TYPES

- DATA ROUTE/DELAY
- FOUR-CYCLE BUS BUFFER
- ZERO COUNTER/ANGLE ESTIMATOR
- DUAL ADDER BIT-SLICE
- SHIFTER
- RAM/PROM/BITE
- NETTER SYSTEM ROUTER

TABLE 8
μSP PARTS COUNT

ELEMENTS	LS TTL IC TOTALS	
	COMMERCIAL IC's ONLY	COM'C'L & 300-GATE ARRAYS
ARITHMETIC		
SCALING	40	11
MULTIPLY	20	8
ADDERS	40	12
CONTROL		
SEQUENCER (1k RAM)	40	14
ADDRESS GENERATOR (1/4k RAM)	40	14
MEMORY		
LOCAL (2k RAM)	25	13
COEFFICIENT (1k RAM) (1k RAM)	20	13
TOTAL:	225	85

SECTION II
ARCHITECTURE OF ELEMENTS

2.1 Orientation

This section develops a computer architecture basis for the design of the micro signal processor elements. Small hardware increases in the memory, arithmetic, control and I/O areas significantly increase computation rate, thus increasing efficiency compared with classic mini computer/micro processor architecture. We foresee that many of the concepts postulated for this μSP will appear in future versions of commercial micro processors.

Raw computing power is only one measure of the worth of any signal processing architecture. This study has analyzed mainstream architecture concepts in detail to judge them by quantitative and qualitative measures including:

- Hardware smallness, showing effects of design efficiency with LSI logic and packaging technology choice
- Hardware simplicity, telling the degree of design match with off-the-shelf building blocks, and the intelligibility of the logic organization
- Hardware assurity, measuring implementation uniqueness and likelihood of interfering with the detail design
- Software thruput, measuring useful computer power for the algorithm mix wanted
- Software simplicity, including coding level, ease of use, and ease of learning
- Software assurity, measuring programming language uniqueness, and likelihood of degrading the software
- Technology transferability, a critical factor for military equipment with their long development cycles compared to semiconductor industry advances

- System efficiency covers the integration aspects not specifically included in any of the above

Our preliminary judgement of some major candidate architectures is summarized in Figure 9. For example, micro processors, netted and/or with multipliers, are shown to be most desirable from the standpoint of hardware size and cost. However, their thruput and software fit to signal processing are not as good as other alternatives.

One architecture alternative shown in Figure 9 implements each task of a big, powerful signal processor with small byte-slices of logic.

	Hardware Smallness	Hardware Simplicity IC	Hardware Simplicity FIT	Hardware Design Assurity	Software Throughput	Software Simplicity HOL	Software Simplicity FIT	Software Design Assurity	Technology Transferable	System Efficiency	Unity Weight Total
Netted Micro Processors	5	5		4	1	1		1	1	1	19
Micro Processor with Multiplier	5	5		4	1	2		1	1	2	21
1 or 2 Bit Byte Slice Pipeline	4	1		1	2	3		4	2	3	20
4 Bit Byte Slice Pipeline	3	2		2	3	3		4	3	3	23
Micro Signal Processor	3	4		3	4	5		3	5	5	32
Mini Signal Processor	1	3		5	5	5		5	4	4	32

Legend: 5 indicates relative best
:
1 indicates relative poorest

Based on Risk, Availability, Experience and Measurable Quantities

Figure 9 - Signal Processor Schemes
(Relative Advantages of Each)

This approach can theoretically take some optimum architecture and trade away speed for reduced hardware. Most of the logic then fits into two or four bit LSI CPU slices. The resulting small number of interconnections often allows PROM or FPLA to replace a collection of cascaded functions. Raytheon understands these techniques because we have used them for fast (1μsec/point) and for small (38 IC) FFTs. End-bit logic problems and conceptual complexity of byte-slice design, however, appear to limit their application to fixed function signal processing. We emphasize that con-

cepts of largr signal processors must be adapted, not adopted, to achieve a μSP.

Also shown in Figure 9 is Raytheon's Mini GPSP, a medium size design with many nice features, but too large for direct application here.

The postulated micro signal processor resulting from this study is also listed in this same table as being small but not smallest; fitting available ICs to a great extent, but not completely; and having final hardware and software detailed by this study.

GP computers, even those using bipolar LSI CPU slices, are dominated in size by their general interfacing structures. Adding a fast multiplier speeds up a micro or mini computer, but still leaves thruput an order of magnitude too slow for signal processing. Not unexpectedly, signal processors are very efficient for arithmetic-oriented tasks. We claim that the μSP has more thruput for its size by at least a couple of powers of two than bigger signal processors. Consequently, large netted μSP systems can be viable, opening up a wider application potential.

The architecture partitioning used successively in Raytheon's large and Mini programmable signal processor was determined to be appropriate here. This partitioning separates the design into elements as follows:

- CONTROL ELEMENTS:
  - SEQUENCER
  - ADDRESS GENERATOR
- MEMORY ELEMENTS:
  - LOCAL MEMORY
  - COEFICIENT MEMORY
  - BULK MEMORY
- ARITHMETIC ELEMENTS
  - SCALING

- MULTIPLYING
- ADDITION

The designs for each of the above elements are derived in the following sections.

2.2 Control and Interface

2.2.1 Overview

The control elements include a sequencer and an address generator. The purpose of the sequencer is to determine both the order of instruction execution and the number of times each instruction is repeated. The purpose of the address generator is to generate the required set of memory addresses for each macro instruction cycle. Communication to the external command and control GP computer is routed through these control elements. Communication includes loading programs, modifying subroutine calling parameters, reading check point values (Build In Test Equipment or BITE), and returning target locations and strengths.

The sequencer element is deveolped in section 2.2.2. It does the bookkeeping for nested macro loops, subroutine linking, and parameter manipulation. A fixed timing cycle is desired to preserve 100% arithmetic and memory utilization for the μSP. This desire conflicts with the utilization of standard GP type components to implement what is basically a GP type function. Hence we have developed the novel concept of using a FIFO type buffer interface between the sequencer and the rest of the control logic. This approach avoids the need for complex look-ahead type sequencer instructions in this sequencer element which is otherwise compatible with commercial micro processor sequencers.

The resulting sequencer design is shown in block form in figure 10. The simulation of the control section is thus based on two processors, the sequencer and the address generator,

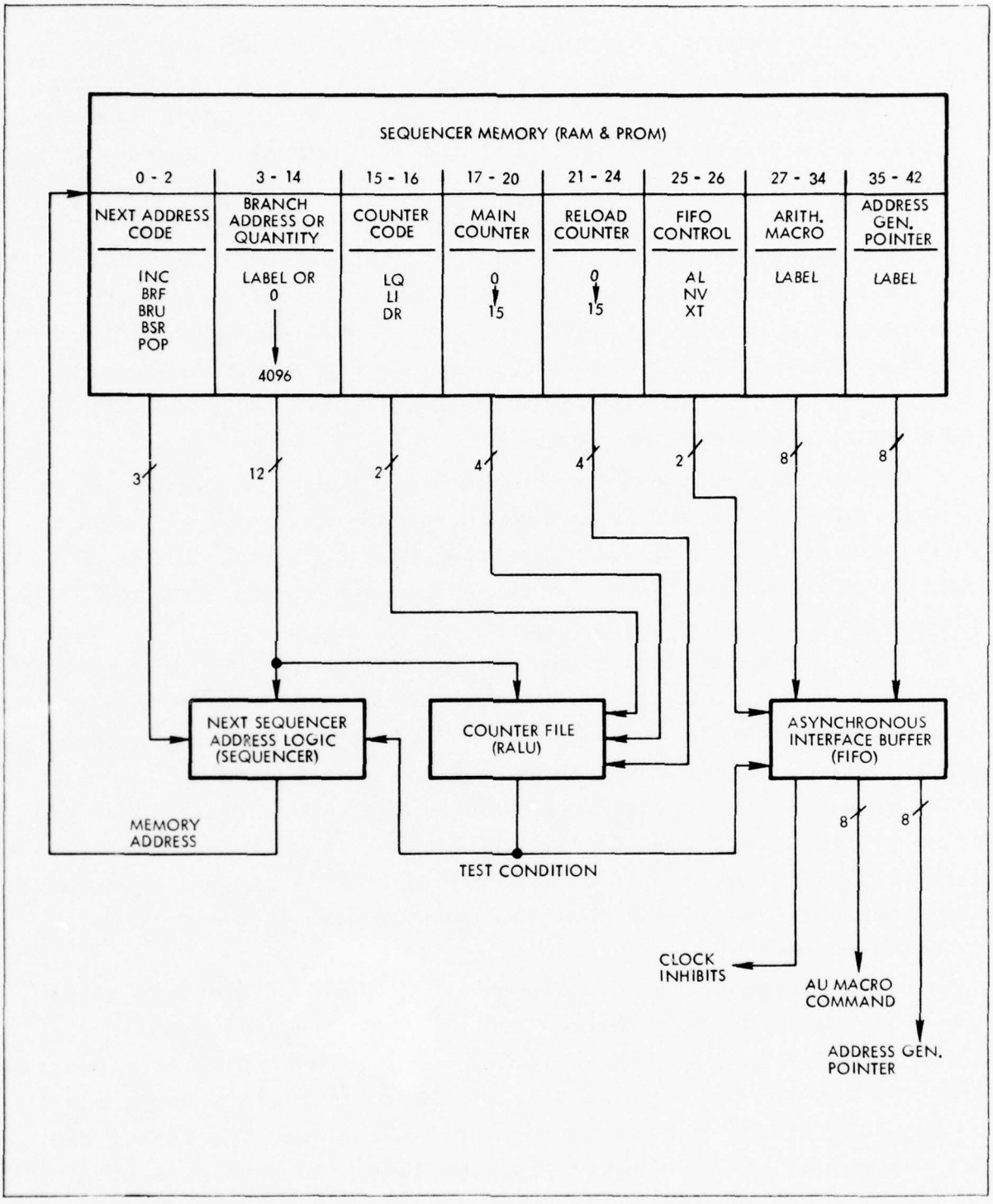


Figure 10 - μSP Sequencer Block Diagram

which are asynchronous to each other. Analytic work was also done on the worst case code, namely manipulating an n-dimensional matrix where each dimension is only two words wide (such as forming the bit-reverse positioning of the FFT output). A sufficient condition to insure 100% utilization of machine thruput is that the clock driving the sequencer logic must operate at least three times that of the macro instruction rate. Since we use four arithmetic clocks to one macro instruction, there can even be a surplus of sequencer cycles available for time-sharing elsewheres. The potential exists for using this surplus to do most of the signal processor driving tasks usually handled by a separate GP computer.

The μSP's sequencer instruction set has been defined for maximum ease of use and to relate to available "sequencer" type IC's. The number of distinct memory control and data fields is thus larger than necessary. This can be pruned down in the next phase when finalizing the design.

The sequencer design can accomodate future generation LSI IC's. We expect that all of the sequencer except for the output data segment could be replaced by the future bipolar single chip μP's. Aside from component reduction and instruction set expansion, this evolution provides potential for incorporating the GP drive computer into part of the μSP design. Such a degree of sophistication is not provided for this generation of μSP, although the sequencer's operation shall be asynchronous to the arithmetic function execution.

The address generator element provides four distinct addresses every macro cycle. These addresses are based on either: a) initialization to a specified location, or b) advancing by a specified increment. One address goes to the coefficient memory, one to the bulk memory and two to the local memories. The latter two are delayed by the arithmetic pipeline length to provide a total of two read and two write addresses for the local memories.

Figure 11 shows the block diagram for the address generator.

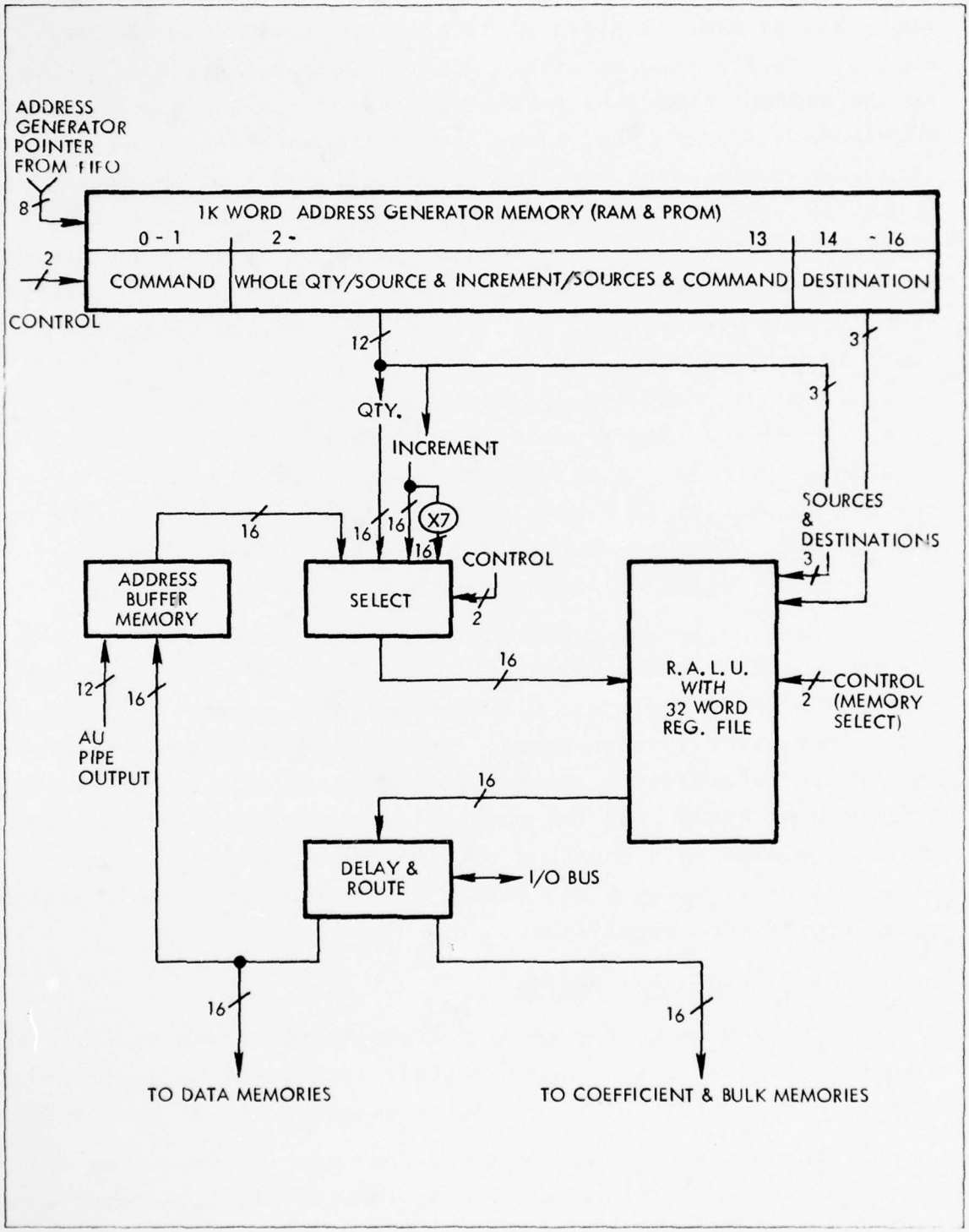


Figure 11 - Address Generator Block Diagram

Note that an address width of 16 bits is provided to the data bulk and coefficient memories. These addresses are controlled by the address generator pointer by way of the address generator microcode. The 16 bits allows for fractional addressing of coefficient tables with less than the full amount of address space. Note also, the use of an address buffer memory that can save either a delayed address or a data component based on an arithmetic pipeline condition. This address manipulation capability provides for data sorting within the μSP. Design details are found in section II.

The control elements both have an amount of program memory. We foresee that a mixture of PROM and RAM can be employed for these purposes. Subroutines can be coded in PROMS while calling sequences, including parameters to be passed, should be kept in RAM. Some parameters can also be passed by reserved register convention in the corresponding elements RALU's.

Several design implementations were generated for these control elements and comparisons made of speed, area and pin totals. A temporary edge in operating speed existed for dropping down to MSI IC's in some areas. Hybrid packaging is a strong contender for achieving volumetric efficiency despite 40 pin IC's. A disturbing trend was the revelation that samples of the newest LSI were slower than promised by significant amounts. The potential of Raytheon's 300 gate array to make an interim design thus becomes more significant.

2.2.2 Sequencer Design

The sequencer for the μSP aims at optimizing speed, real estate, and simplicity. Held constant is a baseline macro coding technique compatible with Raytheon's existing signal processors.

The sequencer block defines the order of execution and the number of repetitions of each arithmetic pipeline macro command and each address generator group command. The sequencer function is very similar to the basic function performed by any

of the bipolar microprocessor sequencer type chips like the 54S482, 9408, and 8X02. These chips however lack the GPSP's multiway jump mechanism based on counter status flags. Therefore arithmetic execution cycles must be skipped some of the time in nested looping. Here, we propose connecting a μP type sequencer to the rest of the uSP through a FIFO buffer. Only a pointer to the arithmetic unit is stored in the FIFO to indicate the instructions to be executed. Sequencing without an arithmetic unit execution cycle occurs by not storing an instruction in the FIFO.

Upon machine initializaiion, sequencer execution starts to fill up the buffer and soon gets ahead of the arithmetic unit's address generator execution. If the buffer is full, the sequencer's clock is shut off. When the buffer is empty, the address generator's clock is shut off. The FIFO allows two different clock rates for writing and reading. Therefor, the sequencer's clock cycle, which affects loading of the FIFO, can be slower than the pipeline clock cycle, which determines the unloading rate of the FIFO. Thus, simple slow sequencer logic works with high speed arithmetic pipelining.

The postulated design provides for incorporating future generation LSI μP parts. All of the sequencer, except a portion of memory and the FIFO and its control, could be replaced by the coming high speed integrated μP's. Instruction repertoire would be expanded allowing more things to happen sooner and possibly with easier coding.

2.2.2.1 Next Address Control

The macro instruction set postulated for the sequencer next address control allows direct branching or subroutine handling and looping by the program stack. The Branch Address/ Quantity field is 12 bits. Only 10 bits will be used for the branch address label while all 12 bits will be available for the quantity field used for loading the counters. For a minimal system this field could be limited to 8 bits.

The instructions are as follows:

Mnemonics	Description	Test	Next Address	Stack	Stack Pointer
INC	Increment	X	Current + 1	N.C.	N.C.
BRF	Test and Jump	True	Current + 1		
		False	Branch Address	N.C.	N.C.
BRU	Jump	X	Branch Address	N.C.	N.C.
POP	Pop and Jump	X	Stack Top	POP'ed	Decr.
BSR	Push and Jump	X	Branch Address	Current+1	Incr.

2.2.2.2 Counter Control

16 RAM words act as indexes or counters, controlled by three mnemonics. The sequencer memory will also contain two fields for addressing these counters; one field will be called the main index field and the other the reload index field. The 3 instructions are as follows:

LQ: Load main index register from the quantity field

LI: Load main index register from the reload index register

DR: Decrement, test the result for zero, and either write the result into the main index register if not zero or write the reload index register into the main index register if zero.

2.2.2.3 FIFO Control

Three mnemonics control the loading of appropriate words into the FIFO. Two different fields are loaded into the FIFO: an 8 bit AU macro label, and an 8 bit address generator pointer for a total of 16 bits into the buffer. The 3 instructions are as follows:

AL: Always load the FIFO

NV: Never load the FIFO

XT: Always load except when the test is true

2.2.2.4 Sequencer Memory

The sequencer memory will consist of a 1K X 43 RAM with structure as follows:

BIT	0 2	3 14	15 16	17 20	21 24	25 26	27 34	35 42
INST	OP CODE	ADR/QUAN	CNT CON	MAIN IN	REL IN	FIFO CON	MICRO	ADGN
M	INC	LABEL	LQ	0	0	AL	LABEL	LABEL
N E	BRF	OR	LI	↓	↓	NV		
M O	BRU	1	DR			XT		
N I	POP	↓		15	15			
C S	BSR							
		4096						

2.2.2.5 Hardware Coding Using Discrete MSI

To accomplish the next address functions required, a design using the Signetics 8X02 sequencer has been done (Figure 2-4). The 8X02's 4 control inputs relate to the op codes chosen as follows.

OP CODE	TEST CONDITION	$S_2S_1S_0$	SSI OUTPUTS TO 8X02 AC 2 1 0	TEST INPUT
INC	D.C.	0 0 0	0 0 1	X
POP	D.C.	0 1 0	0 1 1	X
BSR	D.C.	1 0 1	1 0 0	1
BRU	D.C.	1 1 0	1 1 0	1
BRF (CURR.+1	1	1 1 1	1 1 0	0
BRF (BR. ADR.	0	1 1 1	1 1 0	1

SSI logic generates the above inputs to the 8X02 from the selected bit patterns plus the Test Condition.

The counter control 2 bit op codes are used to control the three state outputs of the adder, the RAM, and the quantity field of the main memory. The op codes will be given a bit assigment to produce minimal logic, and they will be used to control

the three state outputs as follows:

OP CODE	TEST CONDITION	C2 C1	RAM OE-A	ADDER G	QUAN G	RAM OE-B
LQ	D.C.	00	1	0	0	1
LI	D.C.	10	0	1	1	0
DR	0	11	1	0	1	0
	1	11	0	1	1	0

The FIFO control has a 2 bit op code for determining when to load the FIFO with valid data. The FIFO has an input called shift in which is used to load the data. The coding is as follows:

OP CODE	TEST CONDITION	F2 F1	FIFO SI	
NV	D.C.	01	0	don't load
AL	D.C.	00	1	load
XT	0	10	1	load
	1	10	0	don't load

2.2.2.6 Timing

The DR instruction represents the worst case cycle time since this goes through a read-add-write sequence. There are three timing paths to consider. Using data obtained from vendor spec sheets the timing paths are as follows:

Counter RAM access + test NAND logic + 8X02 setup + 8X02 output delay + 256 X 4 RAM access.
=30+15+31+34+50
=160 ns

Counter RAM access + test NAND logic + tristate enable + counter RAM data setup
=30+25+15+30
=100 ns

Counter RAM access + carry bridge adder + tristate delay
+ counter RAM data setup.
= 30+30+22+30
= 112 ns

Hence, the most reasonable cycle time is a clock period of at least 160 ns.

2.2.2.7 Using the 2901

An alternative approach to the next address function is to use the 8X02 with a RAM-adder combination (Figure 13). The control logic to implement the OP CODES defined by $S_2S_1S_0$ will be the same as before. The counter control 2 bit op codes are used to control the 2901 as follows :

			2901 INPUTS			
OP CODE	TEST CONDITION	C_2C_1	I_{876}	I_{543}	I_{210}	C_0
LQ	D.C.	00	2	1	7	1
LI	D.C.	10	2	0	3	0
DR	1	11	2	0	3	0
	0	11	2	0	4	1

The RAM A address is considered the MAIN INDEX. The RAM B address is controlled by a 2——→1 selector which during the read cycle selects the RELOAD INDEX and during the write cycle selects the MAIN INDEX.

The FIFO control decoding is the same as in the discrete MSI design.

The DR instruction represents the worst case cycle time. There are two timing paths to consider which are as follows:

Counter RAM access + test NAND logic + 8X02 setup +
8X02 output delay + 256 X 4 RAM access
= 65+15+31+34+50

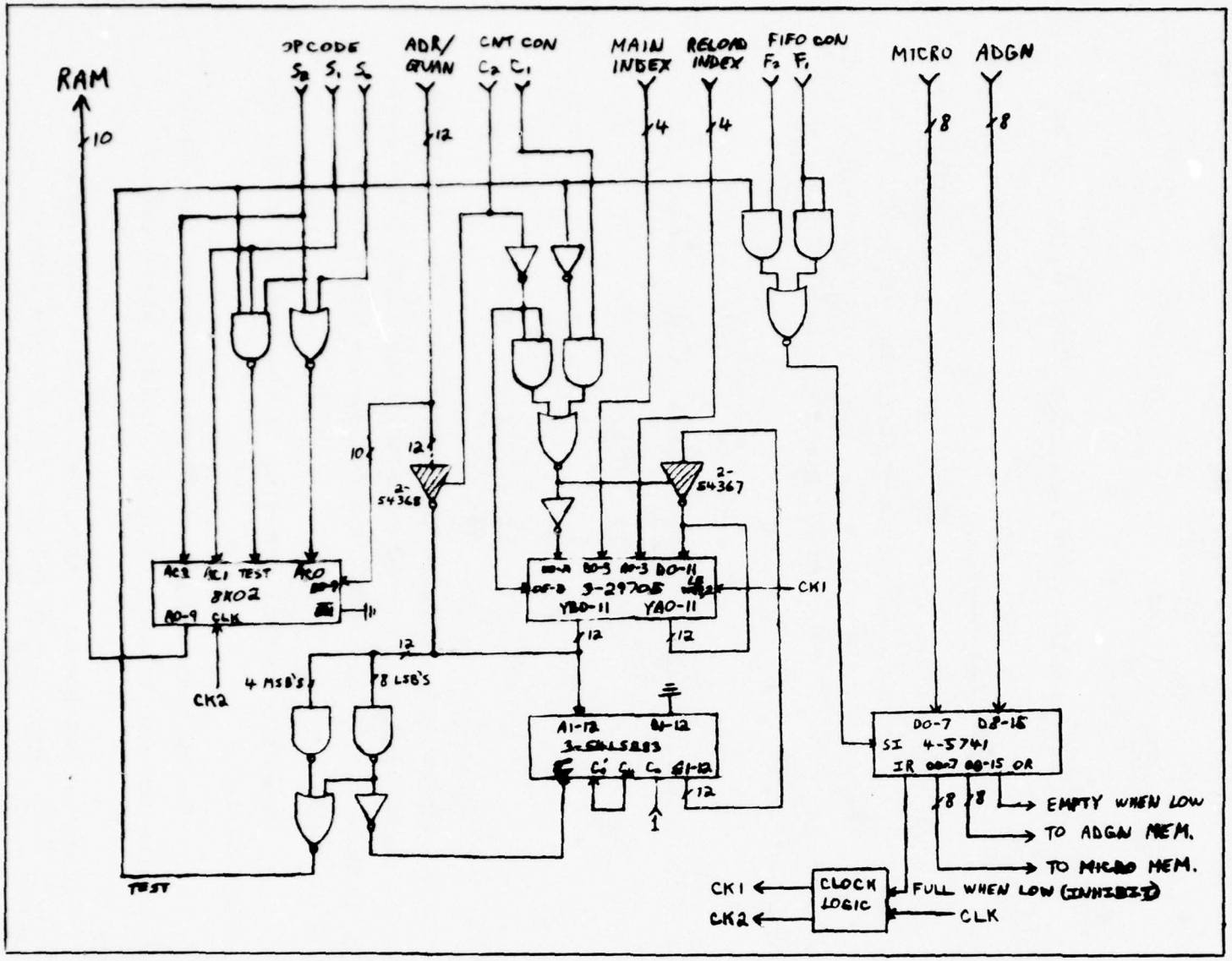


Figure 12 - μSP Sequencer (RAM-ADDER Configuration)

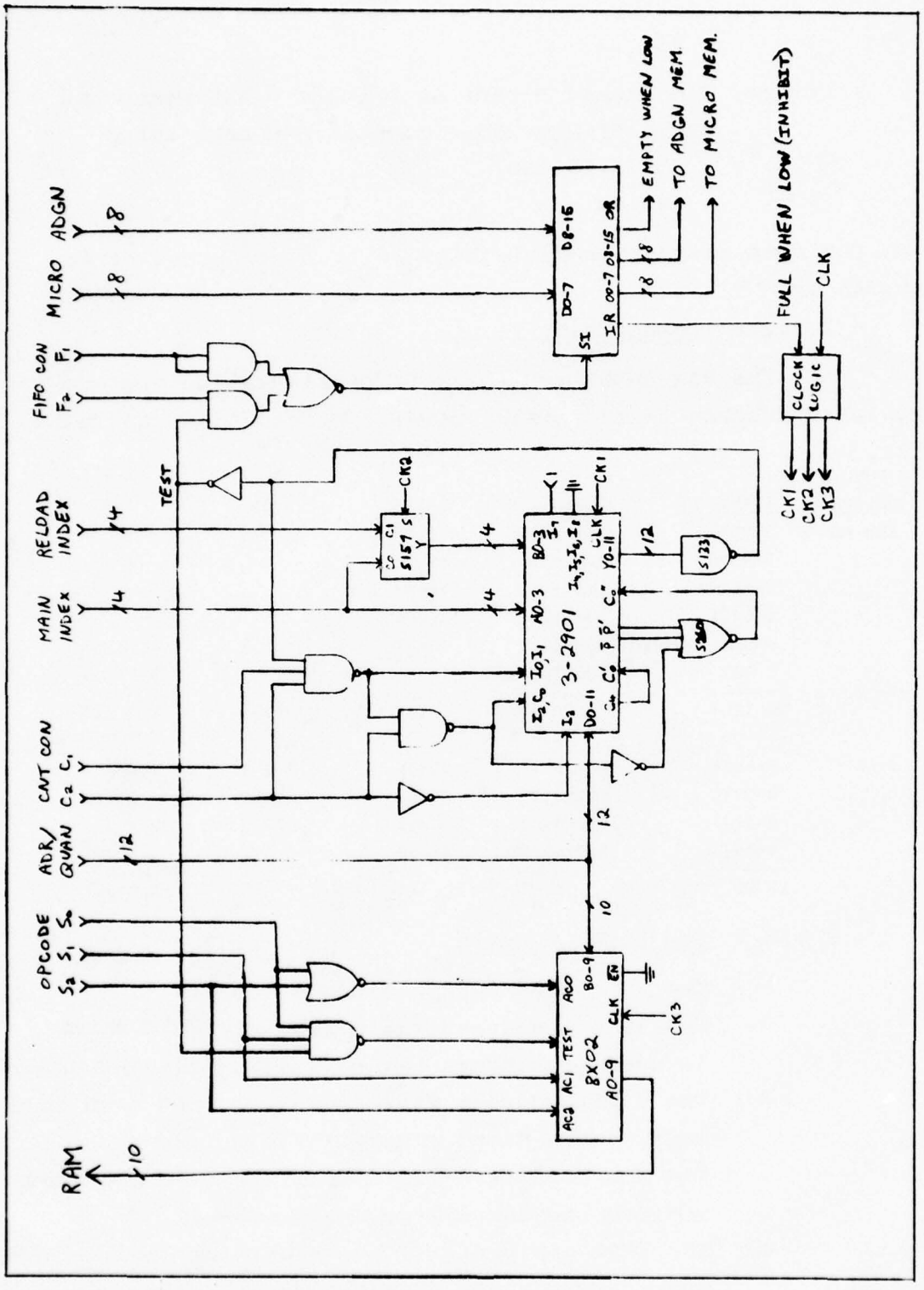


Figure 13 - μSP Sequencer (2901 Configuration)

= 195 ns

Counter RAM access + test NAND logic + ALU setup and carry bridge add + counter RAM data setup
= 65+15+135+30
= 245 ns

Hence, the most reasonable cycle time is a clock period of at least 245 ns.

2.2.2.8 Comparison

The RAM-ADDER and the 2901 configuration are now compared with respect to IC totals, power, package area, and pin count.

DESIGN	PARTS	POWER	PINS	AREA
RAM-ADDER	3-29705	550=1650	28=84	.9 X 4 = 3.6
	1-8X02	650=650	28=28	.24 X 16 = 3.8
	3-LS283	100=300	16=48	
	4-5741	500=2000	16=64	
	4-HEX 3-STATE	325=1300	16=64	
	5-SSI	100=500	14=70	
	20 IC	6.4W	358 PINS	7.4 in^2
2901	1-8X02	650=650	28=28	1.2 X 3 = 3.6
	4-5741	500=2000	16=64	.9 X 1 = .9
	3-2901	1000=3000	40=120	.24 X 8 = 1.9
	4-SSI	100=400	14=56	
	12 IC	6.0W	268 PINS	6.4 in^2

2.2.2.9 Sequencer Outputs

The μSP Sequencer produces 4 outputs:

1. The 10 bit address field from the 8X02 which is used to address the main sequencer RAM memory
2. The 8 bit address field from the FIFO used to address the micro memory.
3. The 8 bit address field from the FIFO used to address the address generator memory.
4. The FIFO empty signal which will be used in the address generator memory control logic.

2.2.2.10 Coding Examples

Figure 14 presents an example of a nested loop coded on the μSP sequencer. Also listed is the resulting sequence of macro instructions and the sequence of address increments.

Programming of a 64 point complex FFT followed by a bit reverse routine has been completed and simulated. This routine takes 28 lines of sequencer code to write, 319 steps to execute, and 258 address generator/macro cycles to execute. Aside from the initial conditions, the pipeline clock worked all the time, doing 192 FFT butterfly macro's, 64 bit-reverse macro's and 8 pipeline flush macro's. More efficient code could be written, but even this effort has shown that the concept of FIFO buffering between sequencer and the rest of the μSP is a viable concept.

2.2.3 μSP Address Generator

The address generator creates the addresses for four memories: data memory 1, data memory 2, the coefficient memory, and the bulk memory. The address generator control memory will be up to 1 K words of 16 bits, and initially all RAM. This 4 cycle machine generates an address for one memory on each cycle. The control memory thus has 256 words for each memory. The FIFO from the sequencer sends an 8 bit address to the MSB's of the ADGN RAM. A 2 bit counter is used to generate the 2 LSB's to the ADGN RAM. Each 8 bit address sent by the FIFO causes the counter to generate 4 counts. Hence, for every 4 cycles the 4 memories are addressed.

2.2.3.1 Instruction Set

The instruction set for creating the addresses will consist of six instructions as follows.

LOC	OPCODE	ADR/QUAN	CNT CON	MAIN IN	RELOAD IN	FIFO CON	MACRO	ADGN
SQ1	INC	2	LQ	15	---	NV	--	--
SQ2	INC	-	LI	2	15	NV	--	--
SQ3	INC	-	LI	3	15	NV	--	--
SQ4	INC	3	LQ	14	---	NV	--	--
SQ5	INC	-	LI	1	14	AL	F1	I1
SQ6	BRF	SQ6	DR	1	14	AL	SAME	Δ1
SQ7	BRF	SQ6	DR	2	15	XT	F2	Δ2
SQ8	BRF	SQ6	DR	3	15	XT	F1	Δ3
SQ9								

MACRO SEQUENCE

F1, F1, F1, F1, F2, F2, F2, F2, F1, F1, F1, F1, F2, F2, F2, F2

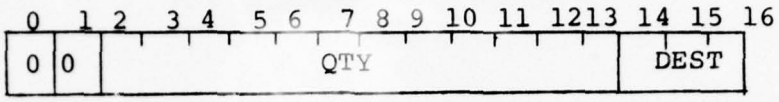
ADDRESS GENERATOR POINTER SEQUENCE

I1, Δ1, Δ1, Δ1, Δ2, Δ1, Δ1, Δ1, Δ3, Δ1, Δ1, Δ1, Δ2, Δ1, Δ1, Δ1

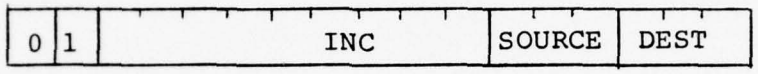
NOTE: MACRO "SAME" WILL KEEP EXECUTING THE PREVIOUS MACRO

Figure 14 - Example of μSP Coding

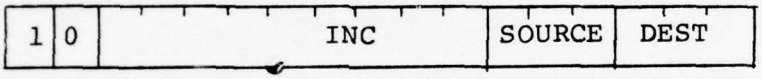
1. LQ - Load Quantity into destination (12 bits plus 4 zeroes)



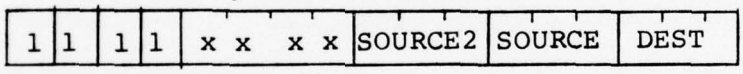
2. IL - Add increment to low part of source and put into destination (extend sign of increment).



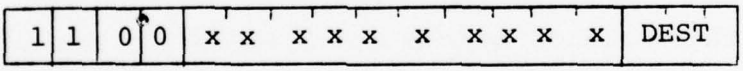
3. IH - Add increment to high part of source and put into destination (zero fill LSB's of increment)



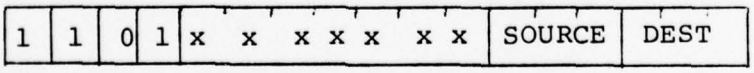
4. SS - Add source to source 2 and put into destination



5. LB - Load address buffer into destination



6. SB - Add source to address buffer and put into destination



A register to register move can be accomplished by an increment of zero using either instruction IL or IH. To add the source to the destination, instruction SS is used by placing the destination in the source 2 field. The LQ instruction is used as follows.

$Q_{11}\ Q_{10} \cdots Q_0\ 0\ 0\ 0\ 0 \longrightarrow$ DEST

where Q_{11} is bit 2 and Q_0 is bit 13 in the instruction.

The IL instruction is used as follows:

$$\begin{array}{l} S_{15}\ S_{14} \cdots S_8\ S_7 \cdots S_0 \\ +\ I_8\ \ I_8 \cdots I_8\ I_7 \cdots I_0 \\ \hline D_{14}\ D_4 \cdots \qquad\qquad D_0 \end{array}$$

Where I_8 (the sign) is bit 2 and I_0 is bit 10 in the instruction and I is a 2's complement number.

The IH instruction is used as follows:

$$\begin{array}{l} S_{15}\ S_{14} \cdots S_7 S_6 \cdots S_0 \\ +\ I_{15}\ I_{14} \cdots I_7 0 \cdots 0 \\ \hline D_{15}\ D_{14} \cdots \qquad\qquad D_0 \end{array}$$

Where I_{15} (the sign) is bit 2 and I_7 is bit 10 in the instruction and I is a 2's complement number.

2.2.3.2 Hardware

There are 32 source and destination registers contained in two 16 word RAM's. Each of the 4 memories has 8 registers assigned to them. The 2 bit counter output is used with the 3 bit source and destination fields to partition the 32 registers into 4 sections as follows:

Q_B	Q_A	RAM Addresses Selected		MEMORY
0	0	UNIT 1	0 - 7	DATA MEM 1
0	1	UNIT 1	8 - 15	DATA MEM 2
1	0	UNIT 2	0 - 7	COEFFICIENT MEM
1	1	UNIT 2	8 - 15	BULK MEM

When the FIFO is empty, a low signal is sent to the ADGN clock logic. This is appropriately used to inhibit the clocks used in the ADGN.

2.2.3.3 Timing

Two designs were done. One used a discrete RAM-ALU configuration (Figure 15) while the other used the AMD 2901 microprocessor (Figure 16). The worst case timing path for each design is as follows.

RAM-ALU:

FIFO access + 1KX4 RAM access + 2 to 1 select + counter RAM access + look ahead add + 2 to 1 select + counter RAM data setup
= 30+50+12+23+20+12+30
= 177 ns

2901:

FIFO access + 1Kx4 RAM access + 4 to 1 select + look ahead add + 2 to 1 select + counter RAM data setup.
= 30+50+20+138+12+30
= 280 ns

However, if a latch is put on the 1Kx4 RAM, and phased clocks are used, then the FIFO and RAM access times become imbedded within the machine cycle time.

Hence, the cycle times are reduced as follows:

RAM-ALU:

= 177-30-50
= 97 ns

2901:

= 280-30-50
= 200 ns

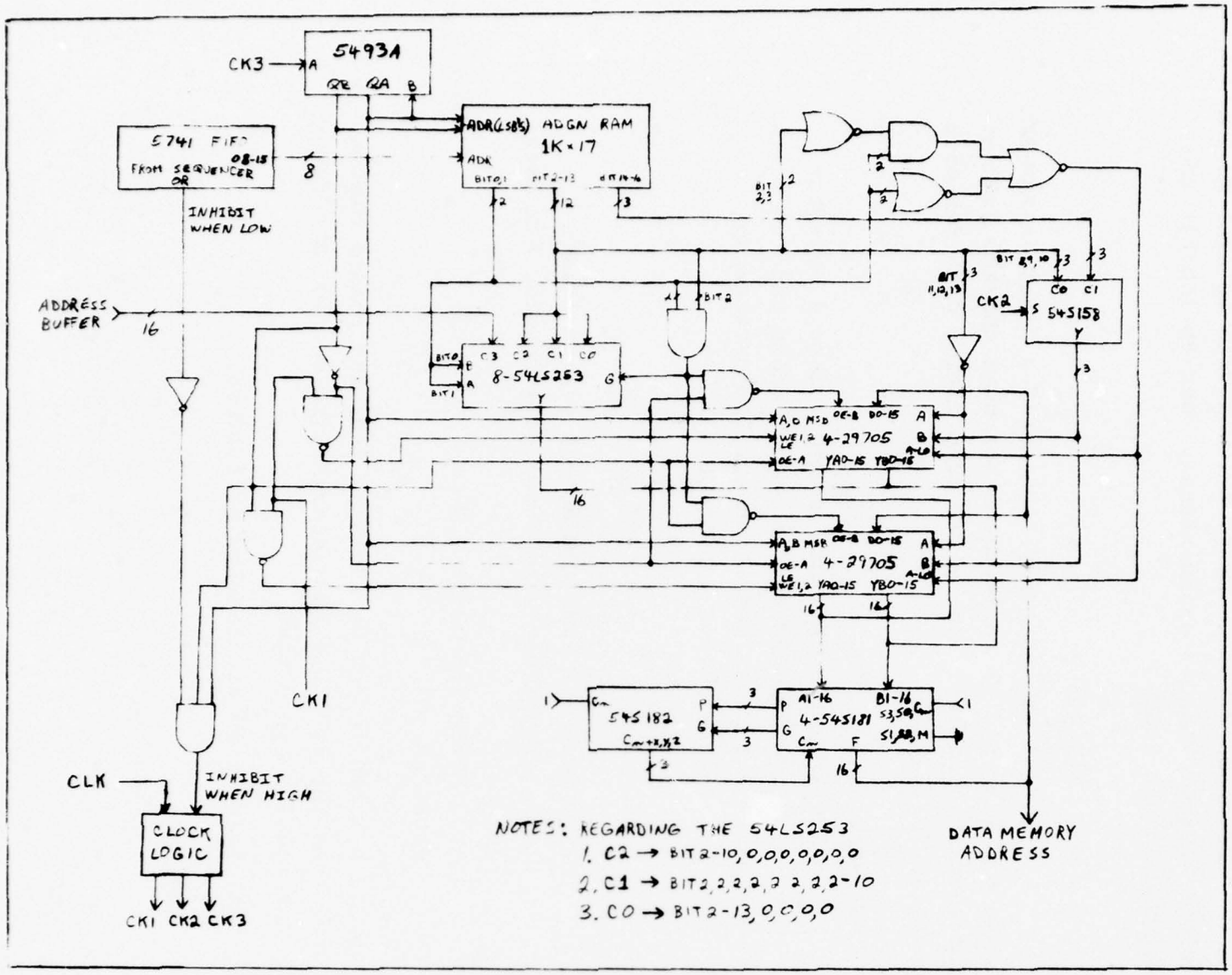


Figure 15 - SP Address Generator RAM-ALU Configuration

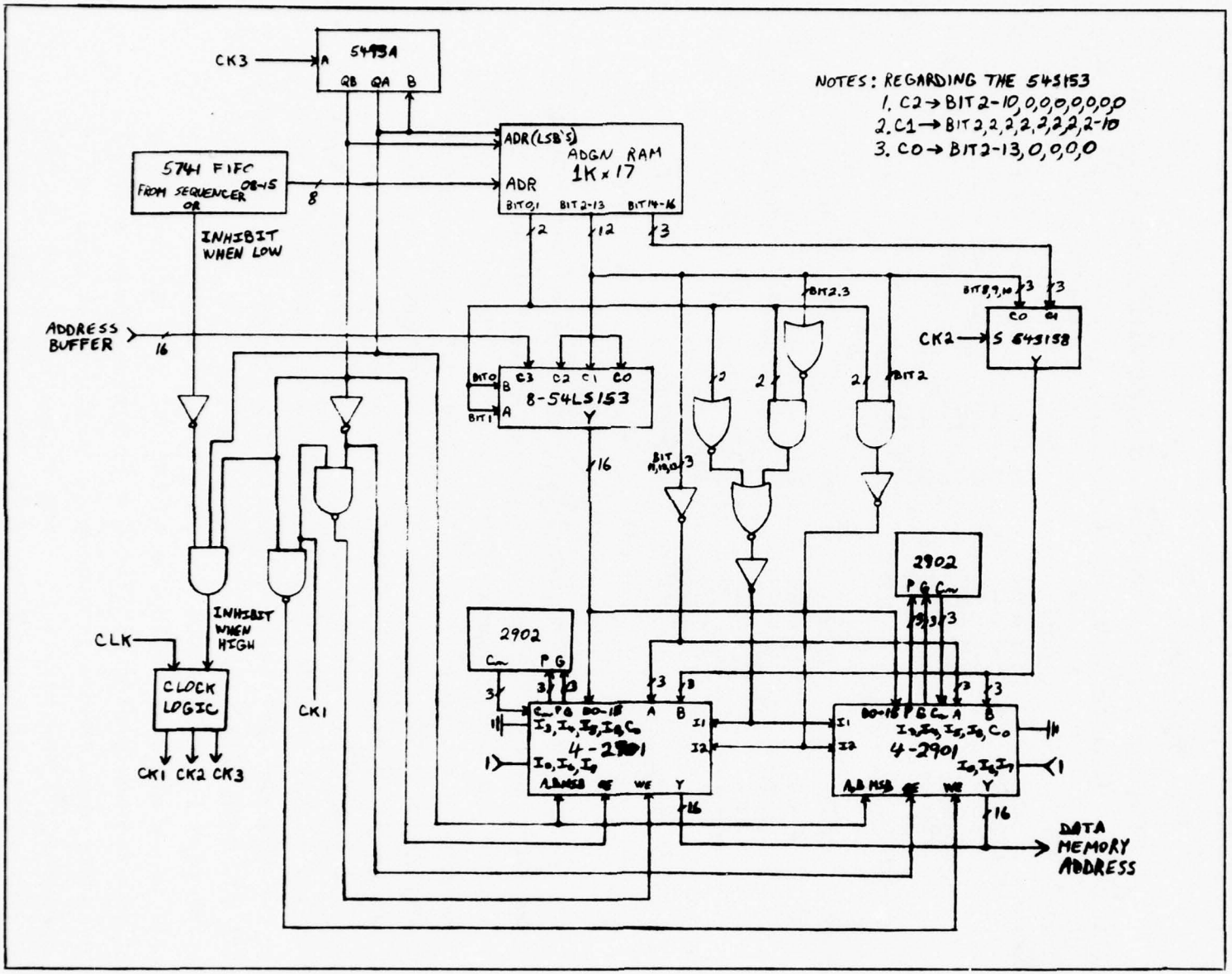


Figure 16 - μSP Address Generator 2901 Configuration

2.2.3.4 Comparison

The two designs are now compared with respect to IC totals, power, package area, and pin count.

DESIGN	PARTS	POWER	PINS	AREA
RAM-ALU	1-54163	150 = 150	14 = 14	.9 x 8 = 7.2
	8-54LS253	50 = 400	16 = 128	.8 x 4 = 3.2
	8-29705	550 = 4400	28 = 224	.24 x15 = 3.6
	4-54S181	700 = 2800	24 = 96	
	1-54S182	350 = 350	16 = 16	
	1-54S158	250 = 250	16 = 16	
	4-SSI	100 = 400	14 = 56	
	27 IC	8.8 W	550 PINS	14.0 in^2

DESIGN	PARTS	POWER	PINS	AREA
2901	1-54163	150 = 150	14 = 14	1.2 x 8 = 9.6
	8-54LS153	35 = 280	16 = 128	.24 x16 = 3.8
	8-2901	1000 = 8000	40 = 320	
	2-2902	350 = 700	16 = 32	
	1-54S158	250 = 250	16 = 16	
	4-SSI	100 = 400	14 = 56	
	24 IC	9.8 W	566 PINS	13.4 in^2

As seen from the two designs presented, the 2901 does not lose very much to the discrete design since 8 discrete RAM's and 4 adders are needed compared to 8 2901's. However, Signetics has announced a two port 32 X 4 RAM. This will reduce the discrete design by 4 IC's and about 100 pins. Further, the eight 4 to 1 selectors in each design could be replaced by a 300 gate array. If the 16 bit address buffer outputs a tri-state signal, and if the 12 bit quantity/increment field is a tri-state, then these two signals are tied together to produce one 16 bit input to the 300 gate array. Bits 0 and 1 from the ADGN control RAM would be used to control the tri-state output enables for the two signals and would also be inputs to the 300 gate array. This would reduce each design by 128-40 = 88 pins. Hence, with all improvements the RAM-ALU design would only use 362 pins while the design would have 478 pins. The bigger RAM's for the RAM-ALU design would also reduce the power by about 2 watts and the area by about 3 in^2.

2.3 Memory

Memory sizes are directly driven by mission parameters, requiring memory to be a modularly expandable item. The memory shapes are determined by the degree of algorithm breakdown employed. We postulate that a minimum of two separately addressed memory units are needed to keep a macro-definable arithmetic pipeline busy. Each of those memories can have data read from it and other data written into it every macro cycle, with no restrictions on allowed sequences of read and write addresses.

Data memory must supply inputs to and take outputs from the arithmetic unit at the latter's fastest operating rate. We postulate that each macro operation shall use no more than the equivalent of two complex data words for input and two for output. Hence, the memory may perform two reads and two writes at distinct addresses during each macro execution time. The ratio between the macro execution times and the memory cycle times will determine the degree of multibucket memory parallelism required. Using a four-cycle ratio avoids any software addressing restrictions, but implies use of fast memory ICs for the working data space. A two cycle ratio requires two separate data memory elements, but lowers the speed, and hence cost, of memory IC's.

The local memory element is blocked out in figure 17. Input and output data paths keep data in the same time sequence, allowing for an arbitrary number of arithmetic pipelines between memory output and input. The order of information on the data paths is arranged to allow operation with either of two types of memories. A memory which can read or write in one clock cycle can serve as the address space for both memory one and memory two. Alternately, a memory which can read or write in two clock cycles can serve as the address space for one data memory unit, with two such memory units required in parallel for system operation. This allows a maximum of flexibility in choosing speed-density for local memories.

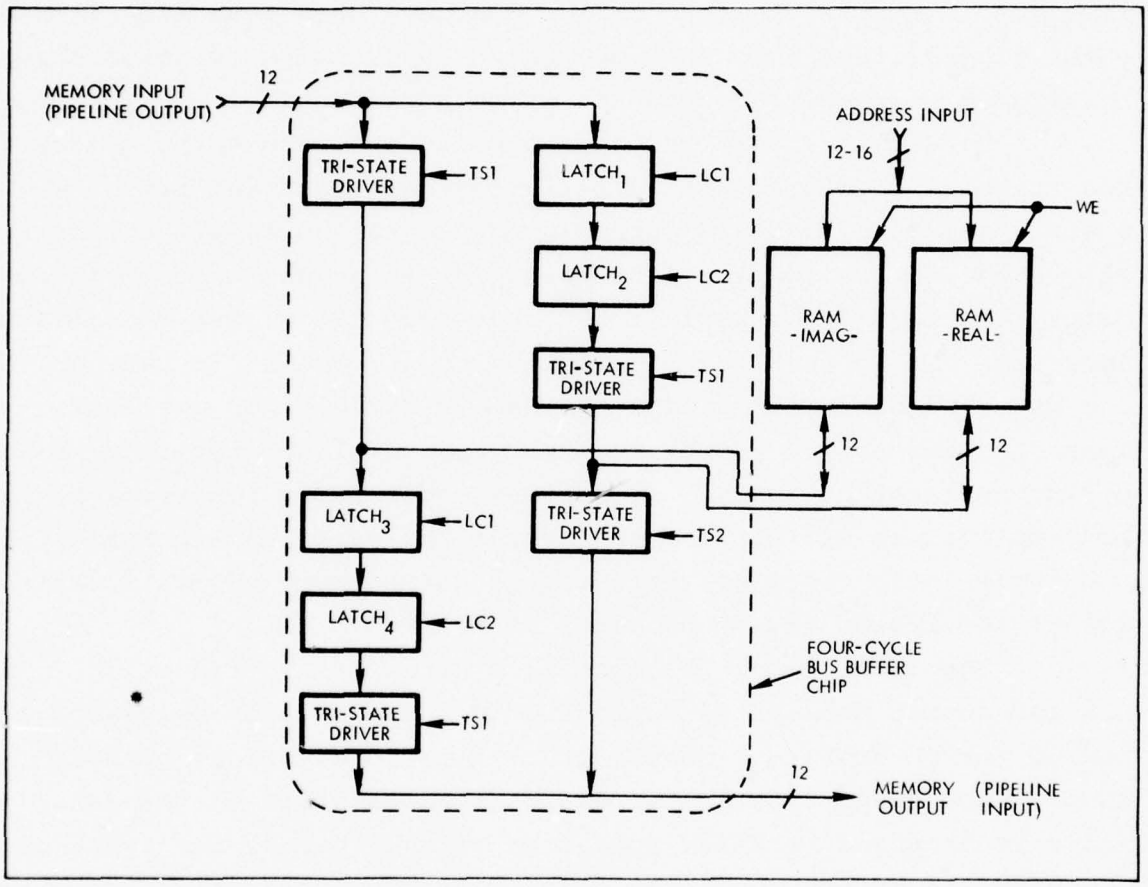


Figure 17 - Micro Signal Processor Local Memory Element

Within the local memory element is a circuit that should be considered for implementation by LSI logic, namely the conversion from a unidirectional input bus to a pair of bidirectional busses and from a pair of bidirectional input busses to a unidirectional output bus. This is essentially a combination of four latches (or registers) plus some tri-state drivers. The number of gates involved as well as the number of pins for a 4 or 6 bits slice is nominal. Such a design would find significant use elsewhere in the μP world compared with today's available selection of bus drivers and buffers.

The coefficient memory element block diagram is shown in figure 18. A combination of RAM and PROM are provided. PROM provides normalized vector tables, such as used in FFTs & Magnitudes, as well as function weightings such as Hamming or Hanning. The RAM provides task dependent parameters such as correction tables for sensors, as well as a path for data to enter the pipeline from the GP I/O bus, RAM vs PROM block selection is done by interpreting the address LSB's as block selectors, with the next bits used for table entry interpolation, and the remaining bits (MSB's) going to memory addressing.

For missions having large storage needs, such as map manipulation, an even slower bulk memory element is desired. We postulate creating such a bulk storage element using denser and slower type IC's, presumably dynamic RAM. The concept is outlined in Figure 19, but will not be detailed in this study due to time and budget limitations. Among the features are:

- Both unidirectional and bidirectional data busses.
- Formatting and address modifying logic for packing/unpacking several short words into one 24 bit format.
- Refresh logic with smarts to avoid refreshing locations that have been addressed recently enough.

The overhead associated with the above features is tolerable when considering the drastic increase in memory packing density (a factor of 4) achieved.

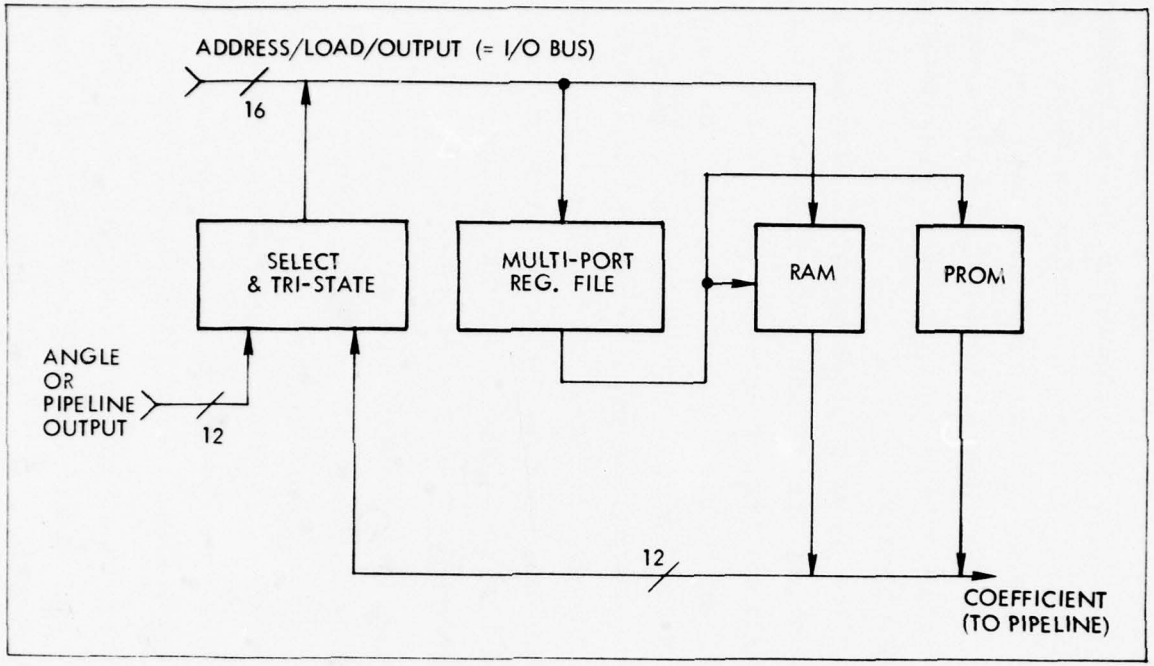


Figure 18 - Micro Signal Processing Coefficient Memory Element

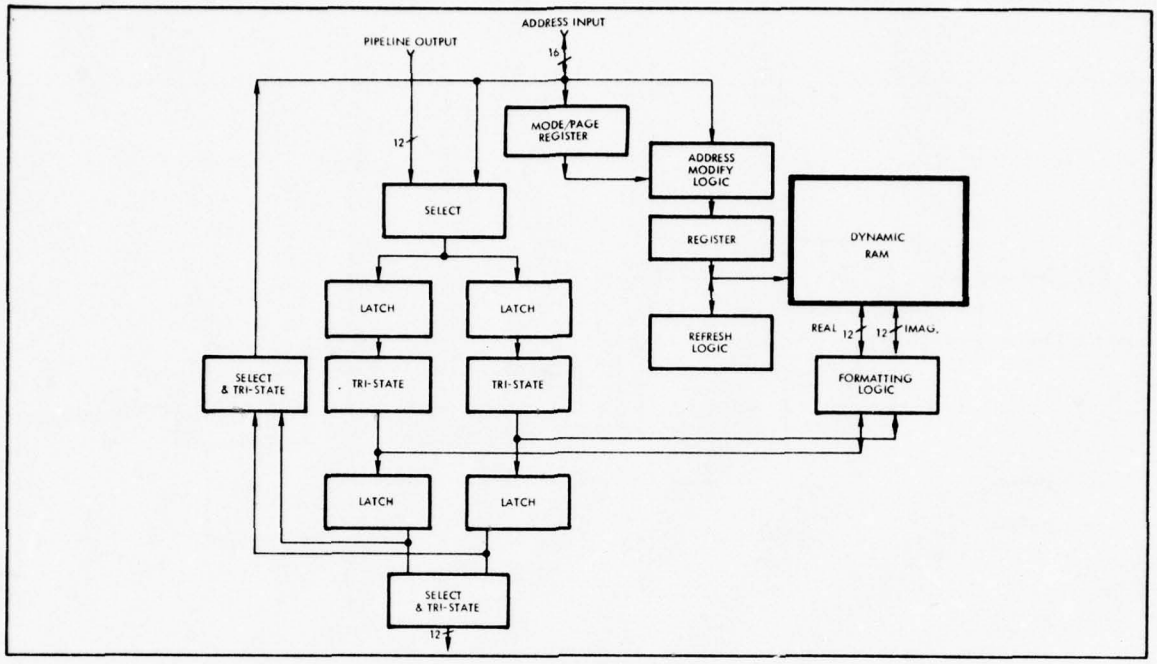


Figure 19 - μSP Bulk Memory Element

Another area where significant improvement is possible is in the combination of RAM and PROM with BITE. In the control elements - the sequencer and the address generator - there must be some RAM, but most code should be in PROM. Similarly, the coefficient memory can be divided into frozen and writable memory. Moreover, all three cases would like to allow the path from the GP I/O bus to write and read such data for BITE purposes. Combining RAM and PROM is now occuring in the slower MOS μP world, and should be considered now for the higher speed μSP environmemt (see figure 20).

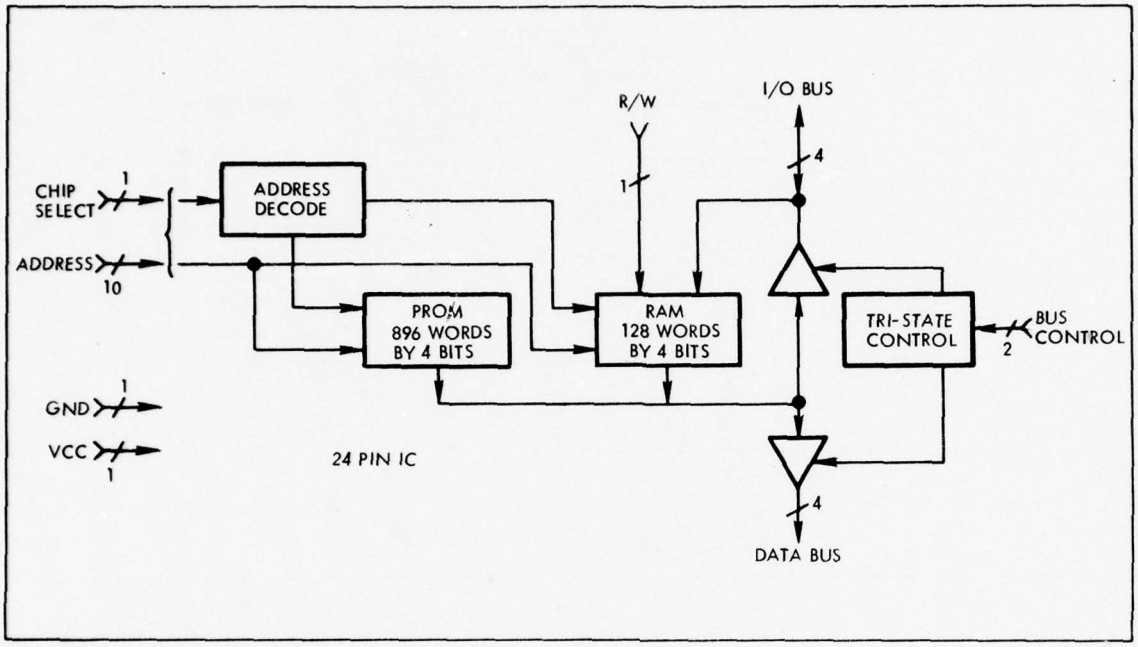


Figure 20 - RAM/PROM/BITE Chip

2.4 Arithmetic Elements

The micro signal processor arithmetic element design is a compromise between small size and large thruput per pass. Multiplier and add/subtractor ratios can be derived from task analysis. Other function capability should be included in hardware where computation alternatives are too slow and ICs involve are few, such as leading zero detection and scaling. To correct a common weakness of GP computers, enough registers must be included for minimizing storage overhead for temporary results.

Organization of registers and arithmetics is key to

achieving thurput efficiency with an easily programmed design. Our mini GPSP experience advocates a compromise between the structured simplicity of pipelining and the unstructured flexibility of microprocessor CPUs. We propose each pipeline stage to have an arithmetic function connected to a multiport register file. Such a pipeline can execute a macro having P times as many operations as hardware by performing P calculations at each stage on its data block before passing the results down the line.

Data will then enter the arithmetic pipeline unit at the same time as the associated command, with timing of macro to micro control bit conversion to match data flow speed. We postulate that the required phased control decoder can be implemented with PROM once the macro command set is defined, just as currently done for GP computers, with room for future macro expansion. The macro commands associated with data leaving the pipeline are similarly generated to facilitate multiple AU configurations.

Control of the arithmetic pipeline is distributed between the sequencer, where a macro command is generated, and the pipleine stages, where a macro acts. One special macro is reserved, called SAME, to indicate that the same macro executed during the last macro cycle should continue. The macro command itself is distributed through the pipeline as a serial bit stream, to minimize pipeline connections, and to allow chaining of pipeline stages. Recommended, but not necessary, is the feeding of the macro output from the pipeline back to the sequencer, to provide for BITE checks. Figure 21 shows the proposed μSP macro decoding logic.

The arithmetic pipeline configuration will consist of three stages; scaling, multiplication, and double adder/accumulator. The design has been conceived to minimize data routing.The expected performance capability is thus limited to a macro command involving no more than 4 multiplies and 8 additions for upward family compatibility reasons, the configurations achievable

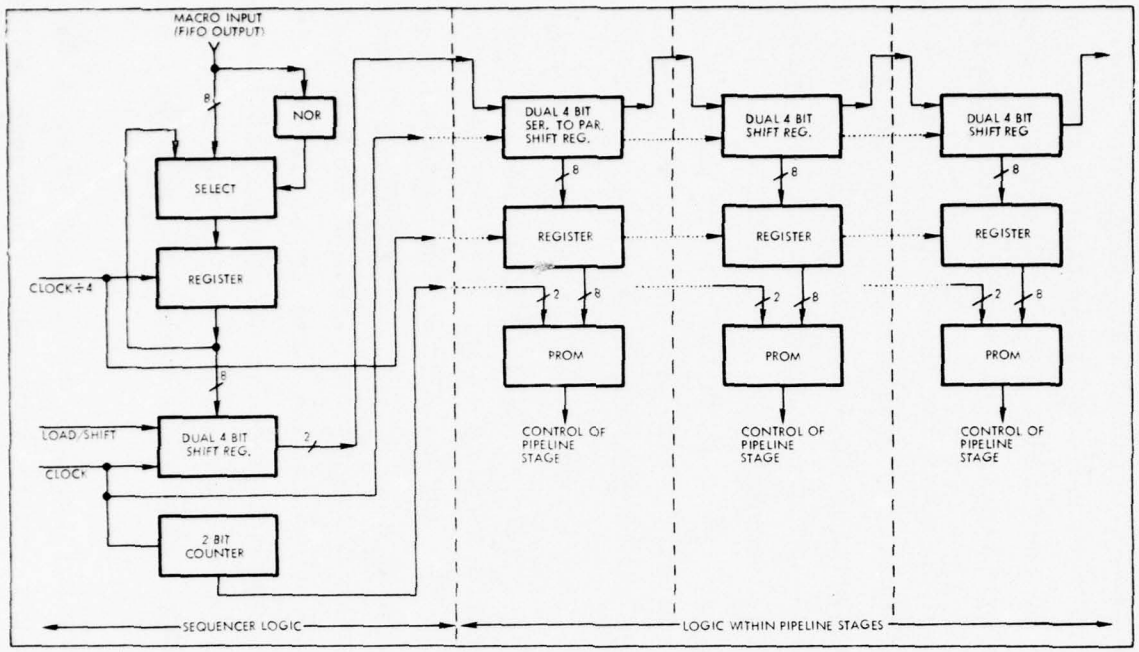


Figure 21 - μSP Macro Decode Logic

with the micro signal processor pipeline should all have a full parallel equivalent. Efficiency of doing double precision calculations particularly with single precision coefficients, is also an attractive feature of this AU design.

As shown in Figure 22, the first stage does scaling, and approximate vector angle determination. The scale factor can be set by counting the number of leading zero's of a special data word, which is usually part of the block floating mechanism. Single precision complex as well as double length real and double length complex formats are accomodated. True floating point can be obtained by doing a block floating point over one word vectors, although this is inefficient in thruput and storage. Figure 23 lists a set of control bit definitions for micro coding the scaling element operation.

Within the scaling element are two candidates for a special implementation, a shifter and an angle estimator chip. The commercial IC world has started to recognize the need for the former, as witness Motorolla's plan for an ECL shift barrel. The angle estimator is a key part of our solution to the magnitude question. Rather than rely on the tradidional "larger plus half smaller" or variations, an approximation scheme, now being patented, can reduce the error from $\pm 6\%$ to below $\pm 1.4\%$. This stops the trade of system performance to save a few gates. Furthermore, more accurate magnitudes are possible by iterative processing. Finally this angle estimator chip simplifies monopulse computations.

The second pipeline element is the multiplying unit shown in Figure 24. This element does 12 by 12 multiplications with possibility of saving and working with all 24 result bits if desired. Data ordering for the third stage is actually performed within this second stage, in the order of multiplication. One of the two inputs to this element comes from the first stage, while the other comes from the coefficient memory. Because of the

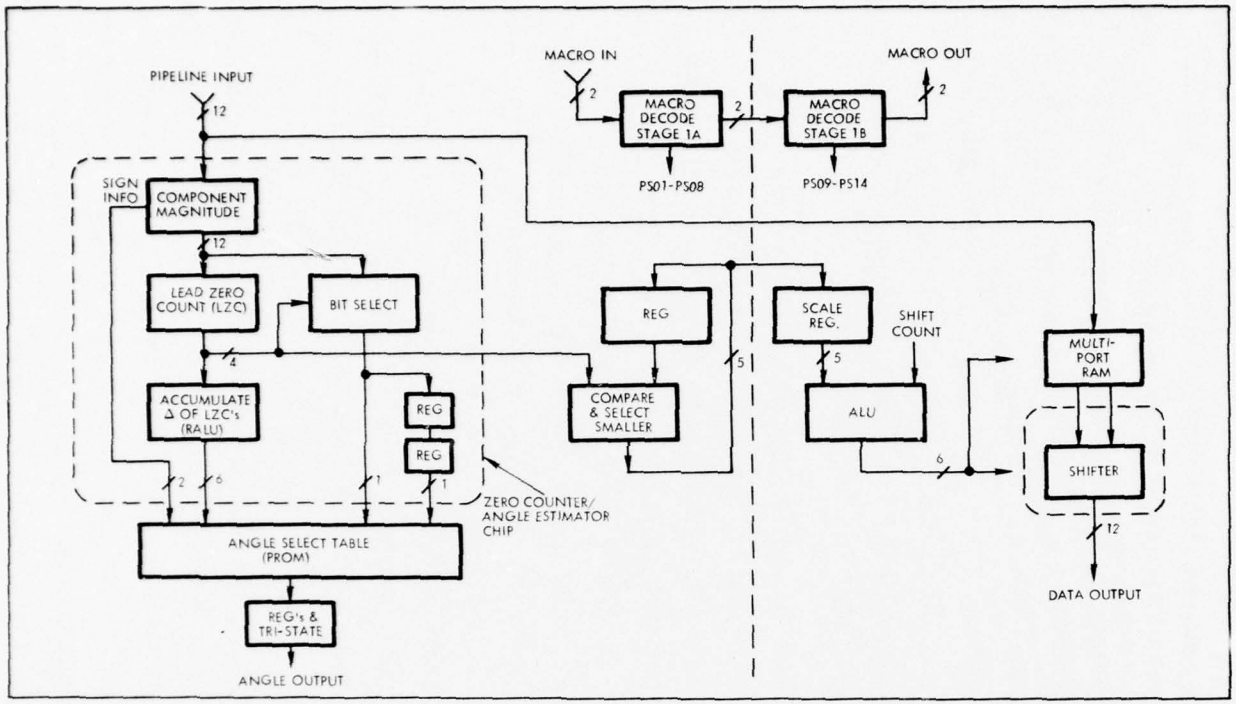


Figure 22 - Scaling Element

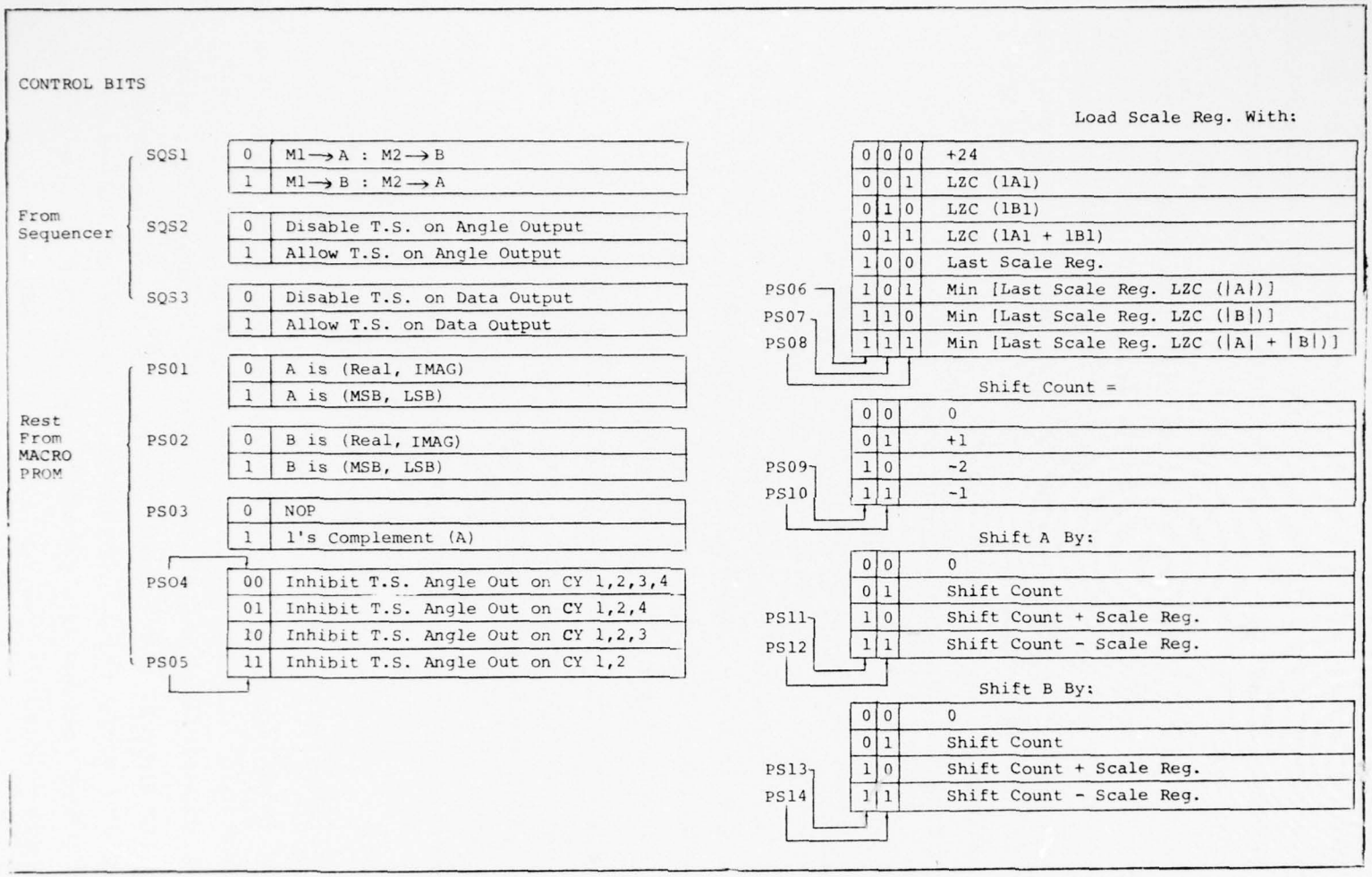


Figure 23 - Pipeling Scaling Element

flexible routing empolyed, the coefficient memory path can be used as a means of introducing fixed constants and even addresses into the data stream. Figure 25 lists the control bits which must be specified for each clock cycle of operation.

A data route/delay function has been outlined in Figure 24. Such a chip design has usefulness within the μSP in the multiplier, the adder and the address generator elements. Delay sizes do not exceed 32 and are typically in the 8-16 range. This chip type is thus not intended to replace RAM, as done in RCA's CMOS FFT chip set. We see the real problem as combining the switching of input and output paths efficiently in the presence of read while write RAM.

The third pipeline element contains two ALU's, accumulator registers, and a peak detector as shown in Figure 26. This stage does the additions required to complete a complex multiplication as well as the "butterfly" type combinations. Both single and double length calculations are possible. A listing of possible adder element micro control bits are shown in Figure 27.

Implementation of the arithmetic pipeline doesn't fit the 2901 type RALU design very well. The quantity of working registers needed is typically 4 rather than 16, and only a few arithmetic functions are required. The only available IC's for implementing the scaling requirement are the low density a priority encoders and 4 bit shifters, and perhaps a multiplier used wastefully as a shifter. Hence, fertile ground exists for IC type improvement in this area.

The above design for the multiplier and adder element has favored functional simplicity at a cost of extra micro control complexity. A viable alternative is to combine a long adder with the multiplier element to finish the complex or double length multiplications there. Such an alternative has been used before at Raytheon, and reduces the flexibility required in the final pair of adders. However, we opt here for having a more

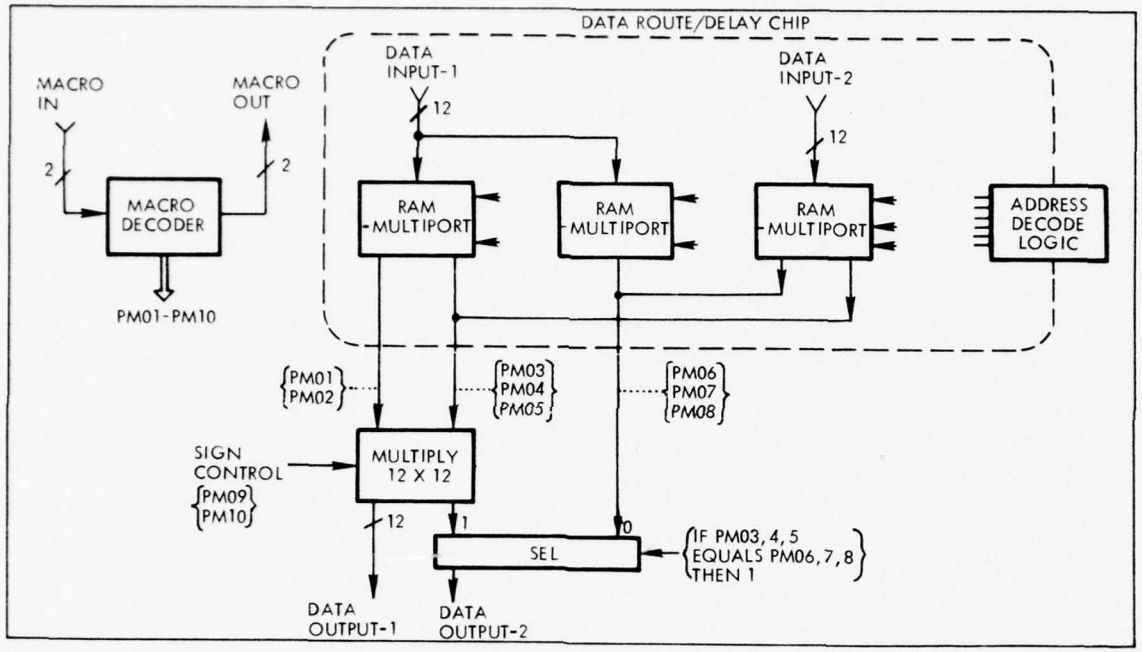


Figure 24 - Multiplying Element

	PM01	PM02	PM03	PM04	PM05	PM06	PM07	PM08
INPUT - 1 : A - REAL	0	0	0	0	0	0	0	0
INPUT - 2 : B - REAL	0	1	0	0	1	0	0	1
INPUT - 1 : A - JMAG	1	0	0	1	0	0	1	0
INPUT - 1 : B - JMAG	1	1	0	1	1	0	1	1
INPUT - 2 : A - REAL			1	0	0	1	0	0
INPUT - 2 : B - REAL			1	0	1	1	0	1
INPUT - 2 : A - IMAG			1	1	0	1	1	0
INPUT - 2 : B - IMAG			1	1	1	1	1	1

PM09

0	MULTIPLIER SIGNED
1	.. NOT ..

PM10

0	MULTIPLICAND SIGNED
1	.. NOT ..

Figure 25 - Multiplying Element Macro Bits

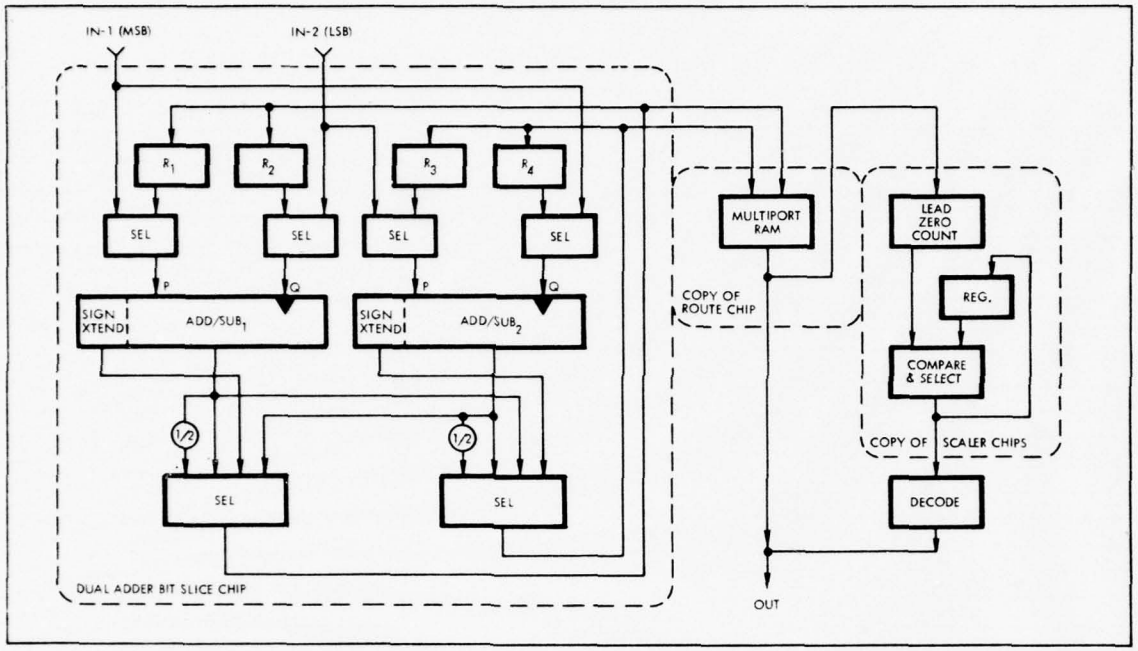


Figure 26 - μSP Adder Element

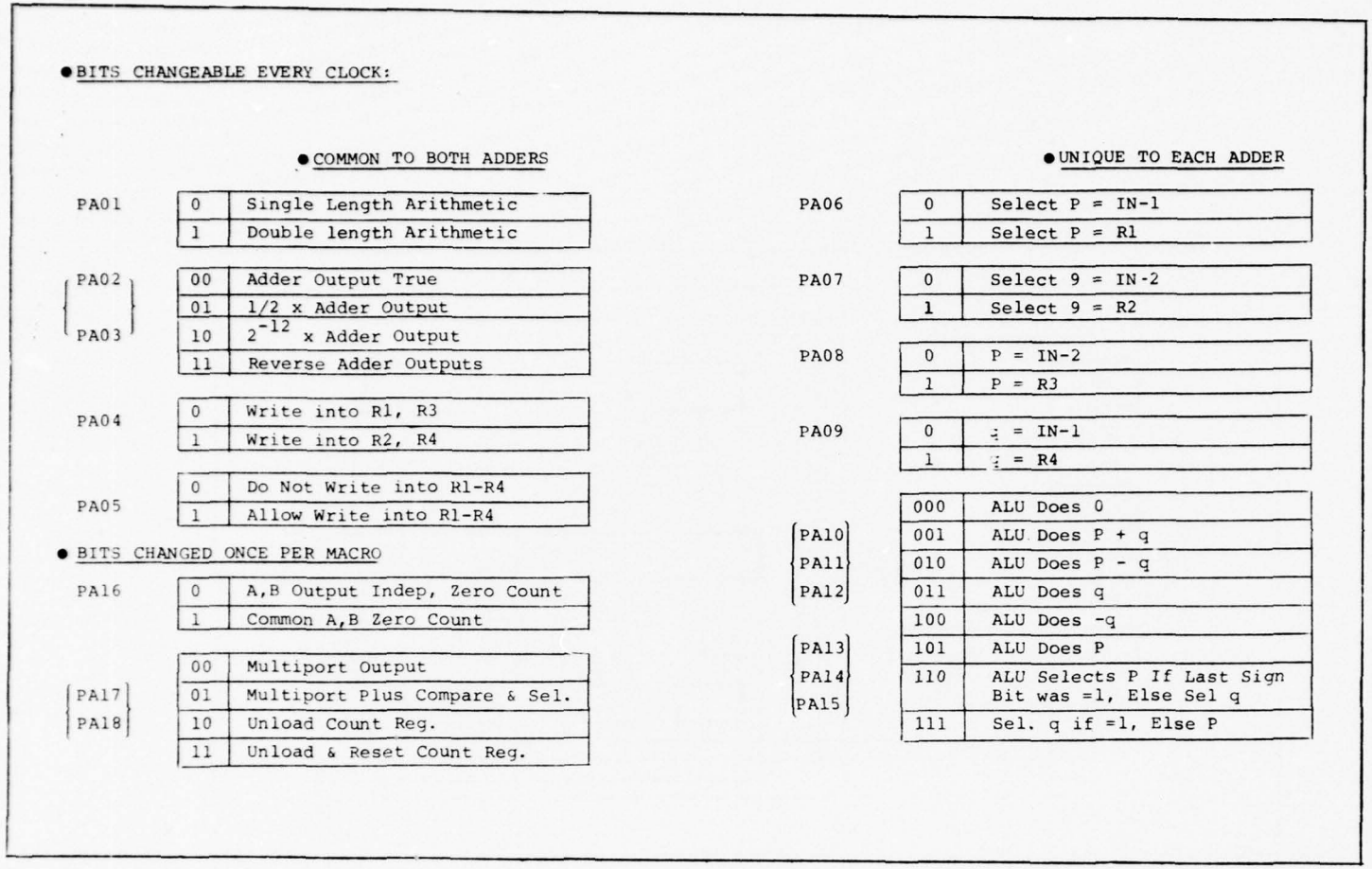


Figure 27 - Adder Element Micro Bits

flexible final stage in order to accomodate the strong tendency seen to specialize programmable signal processors by adding an extra CFAR-type post-processor into the arithmetic pipeline. We expect to be able to accomodate many members of the "CFAR of the month" algorithm club with the three element pipeline. For those cases where significant improvement is possible by giving in, then we need add only another identical adder element.

2.5 Arithmetic Macros

From the analysis which is presented in section 3, we have identified the following list of basic signal processing macros:

Vector normalize
Vector scale and add
Sum of vector elements
Vector dot product
Peak detect
Complex vector multiply
Accurate magnitude
Correlation
Complex FFT
FIR
IIR
Sliding window sum

From the above list, most of the macros can be derived in a very straight forward manner, because the functions fall obviously out of the element definitions. For example, accurate magnitude happens by a complex word generating an angle in the scaling element, which pulls out the appropriate unit vector from the coefficient memory, which rotates the complex word through multiplication and addition.

The operation of the FFT is a little more complicated and is thus shown in Figure 28. Note that the adder pair is used

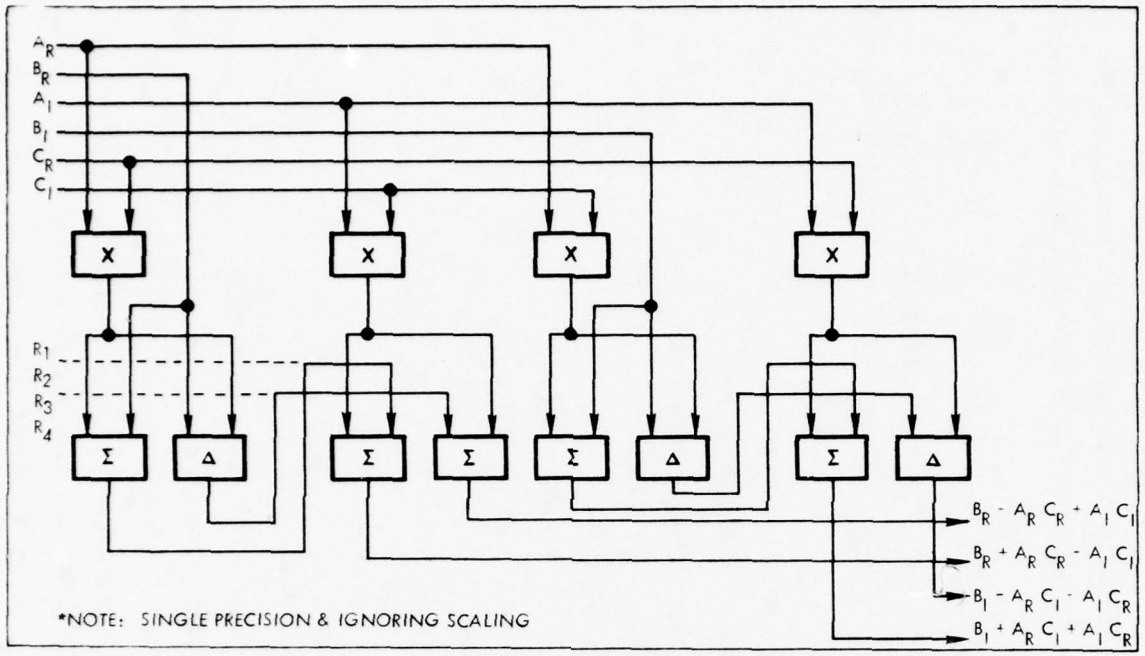


Figure 28 - Macro for FFT*

wastefully, doing 8 adds when only 6 are required. This inefficiency is is illusionary because during the 4 clock cycles 2 adds would otherwise be unused. Extension of this concept to definition of the set of three macros required for double length (24 + j 24 bits) word FFT's, is shown in Figure 29. Note that the locations of the data in and date out are properly lined up to allow only two local memory addresses per macro cycle, even with double length data.

The IIR type filter is shown partitioned in Figure 30 into the three macro's described in Figure 31. The first macro beats the input point with the desired oscillator, and leaves the result in the accumulators. The local memory is used to store all filter residues. Thus a pair of zeros are computed by combining two delay element outputs with the accumulator hold over. Similarly, the macro for the pole pair ends up storing two complex numbers. In this fashion the number of filter poles and zeros is not limited by the amount of storage contained within the arithmetic pipeline.

Partitioning of the CFAR algorithm for the uSP arithmetic pipeline is shown in Figure 32. Two macros are involved. The first forms a magnitude and a sliding window accumulation. The second macro does the actual scaling of threshold value and comparison against the target value. Use of the adders for condition testing can then tie into saving the addresses of the selected bins in the address generator.

2.6 System Considerations

A simple connection of μSP elements into a system with one of each type element has already been presented. This section considers variations on the interconnection possibilities.

Simple paralleling or pipelining of elements is always possible, such as shown in Figure 33. Conversely, selected elements can be employed to make a compact hard-wired function, such as the FFT unit shown in Figure 34.

MACRO #	DATA IN:	MULTPLIER OP's:	ACC.'s HOLD:	DATA OUT:
1	A MSB, LSB IMAG B MSB, LSB REAL	A_I^L X COS θ A_I^N X COS θ B_R^L X 1 B_R^M X 1	MSB LSB $(B_R + A_I \cdot \cos\theta)$ MSB LSB $(B_R - A_I \cdot \cos\theta)$	_____ _____
2	A MSB, LSB REAL B MSB, LSB REAL	A_R^L X SIN θ A_R^M X SIN θ A_R^L X COS θ A_R^M X COS θ	MSB LSB $(A_R$ X COS $\theta)$ _____	MSB LSB $(B_R + A_R \cdot \text{SIN} - A_I \text{ COS}) \pm B_R'$ MSB, LSB $(B_R - A_R \text{ SIN} + A_I \text{ COS}) \pm A_R'$
3	A MSB, LSB IMAG B MSB, LSB IMAG	A_I^L X SIN θ A_I^M X SIN θ B_I^L X 1 B_I^M X 1	_____ _____	MSB, LSB $(B_I + A_R \text{ COS} + A_I \text{ SINE}) \pm B_1'$ MSB, LSB $(B_I - A^R \text{ COS} - A_I \text{ SINE}) \pm A_J'$

Figure 29 - Double-Length FFT (3 Macro's per B"T"Fly)

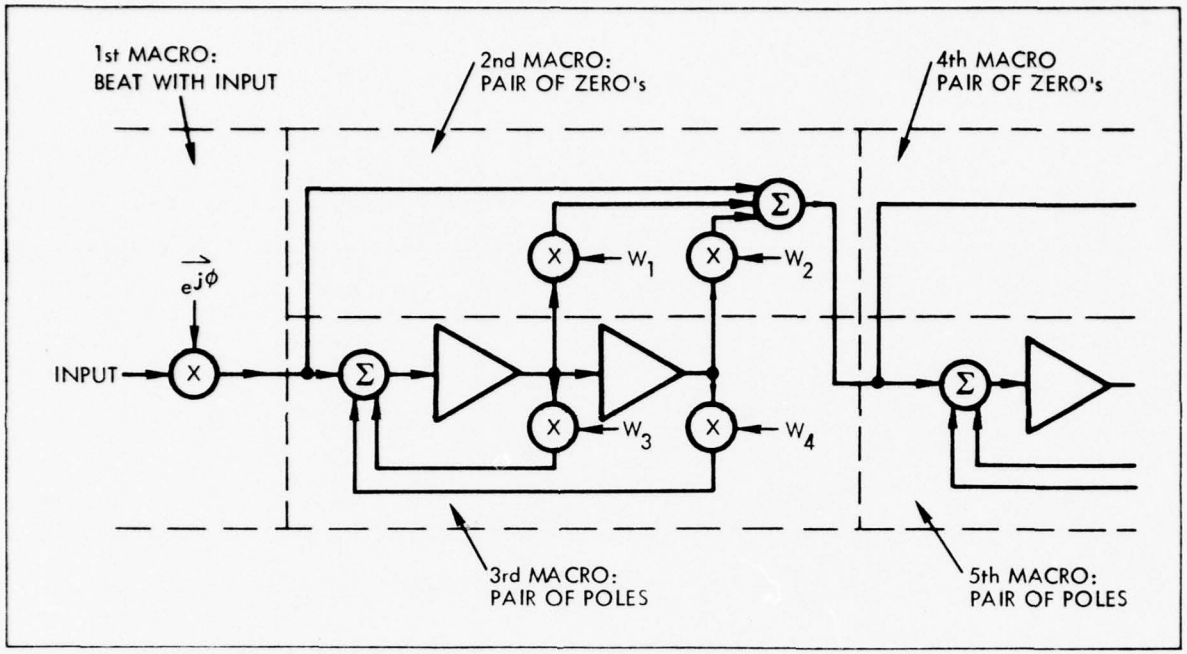


Figure 30 - IIR Filter Partitioning

MACRO	DATA IN:	MULTIPLIER OP's:	ACC.'s HOLD:	DATA OUT:
BEAT WITH INPUT	A_R	$A_R \cdot \text{SIN}\,\emptyset$	$A_B' = A_R \text{ SIN} - A_I \text{ COS}$	
	A_I	$A_I \cdot \text{COS}\,\emptyset$	$A_I' = A_R \text{ COS} + A_I \text{ SIN}$	———
	———	$A_R \cdot \text{COS}\,\emptyset$	———	
	———	$A_I \cdot \text{SIN}\,\emptyset$	———	
PAIR OF ZEROS	P_R	$P_R \cdot W_1$	$A_R'' = A_R' + P_R W_1 + Q_R W_2$	A_R''
	P_I	$P_I \cdot W_1$	$A_I'' = A_I' + P_B W_1 + Q_I W_2$	A_I''
	Q_R	$Q_R \cdot W_2$	A_R'	———
	Q_I	$Q_I \cdot W_2$	A_I'	———
PAIR OF POLES	P_R	$P_R \cdot W_3$	A_R''	Q_R
	P_I	$P_I \cdot W_3$	A_I''	Q_I
	Q_R	$Q_R \cdot W_4$	———	$A_R' + P_R W_3 + Q_R W_4$
	Q_I	$Q_I \cdot W_4$	———	$A_I' + P_I W_3 + Q_I W_4$

Figure 31 - IIR Filter Macros (3 Types)

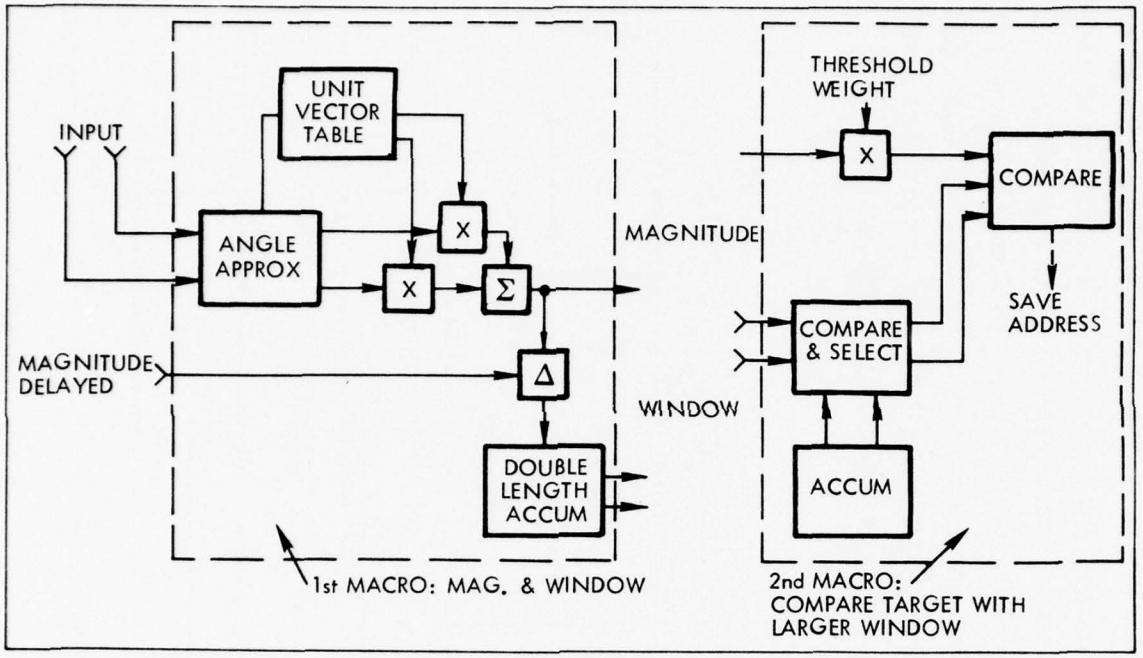


Figure 32 - CFAR Algorithm Partitioning

More complex netting of systems is also possible, but this deserves more than the usual lip service. Particular consideration should be given to the problem of routing all the high speed data paths in a netted system of parallel and pipelined elements. For this , we propose the netted system router box depicted in Figure 35. Serial Transmission is employed between stages to minimize the number of wires entering each identical unit, while still allowing arbitrary interconnections for fault tolerant operation. This concept of a corner-turning memory has equivalent designs for both analog and digital switching interfaces.

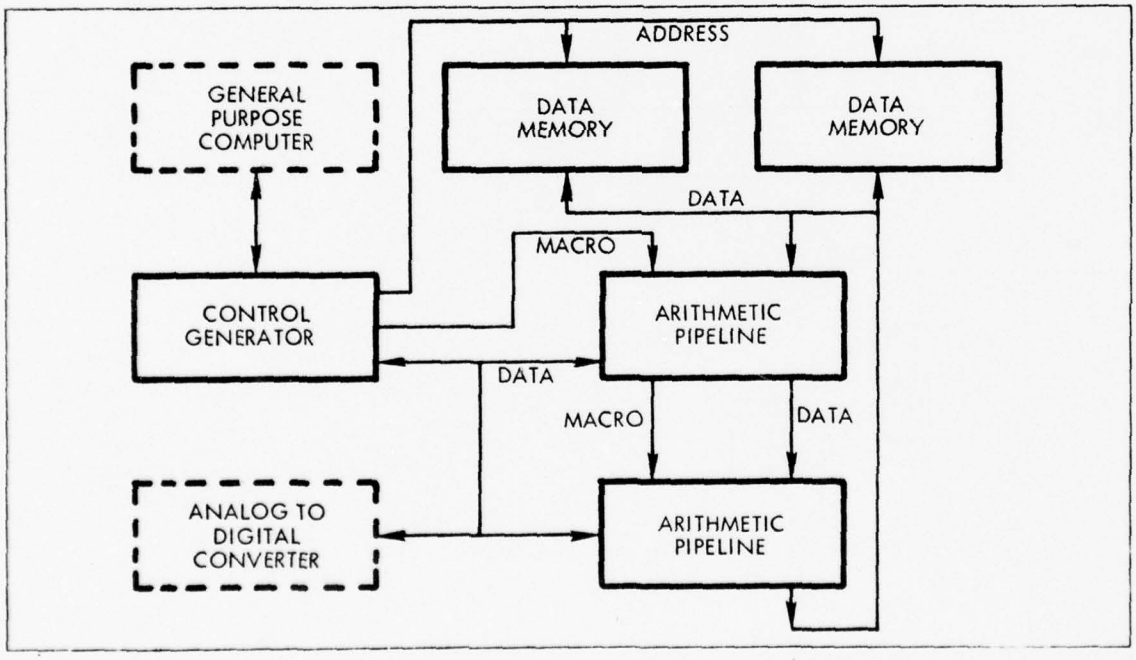


Figure 33 - System Example of Micro Signal Processor

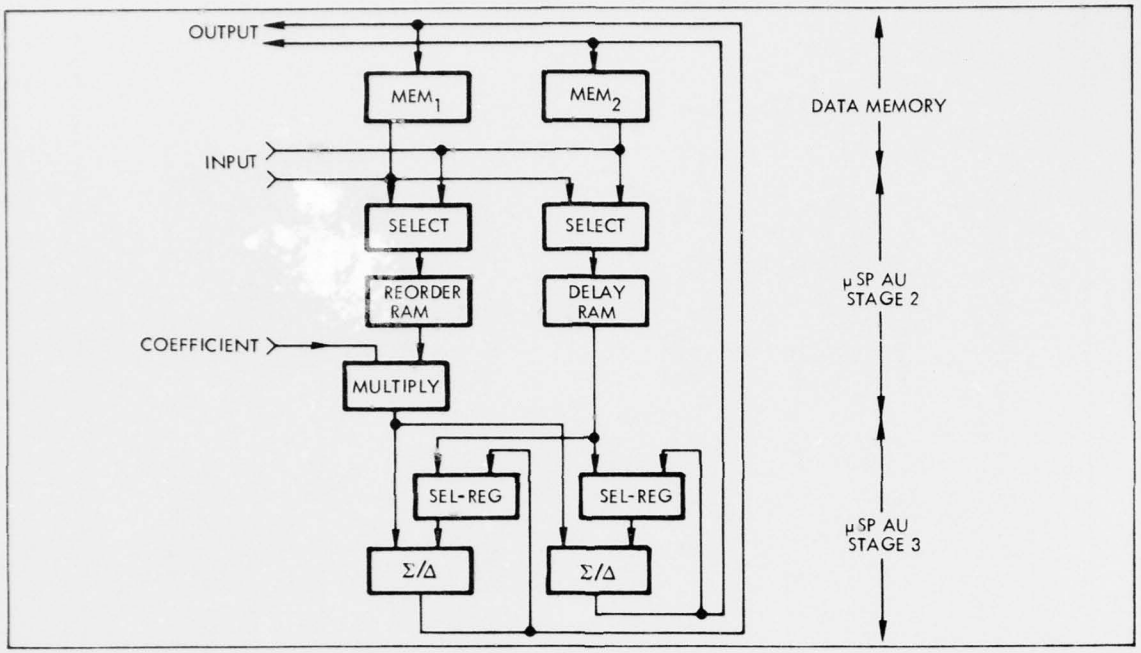


Figure 34 - Hardwired FFT Using μSP Parts

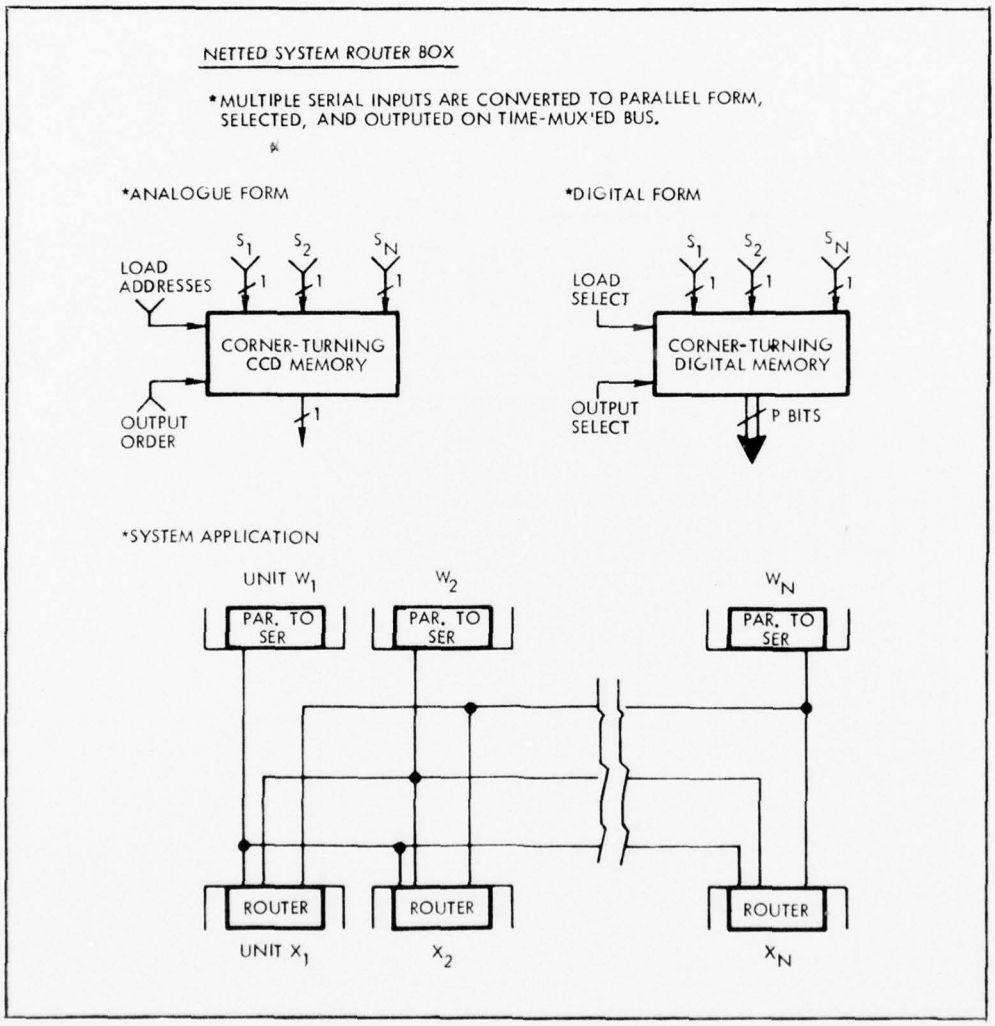


Figure 35 - Netted System Router Box

At some point, netting small μSP's should give way to use of a more powerful unit, here called a MAXI SIGNAL PROCESSOR. We postulate that there should be, for the same technology implementation, a factor of four difference in thruput and size between these two units. Certain compatibilities can be preserved between the two, leading to the concept of a family of programmable signal processors. The following lists some of the ways to grow from the μSP design to this Maxi μSP.

- SEQUENCER:
  - * DOUBLE-BUCKET ON TWO UNITS
  - * LOOK-AHEAD ON COUNTERS
- ADDRESS GENERATOR
  - * PARALLEL TWO OR FOUR UNITS
- DATA MEMORIES
  - * SEPARATE MEMORY ONE AND TWO
  - * DOUBLE BUCKET EACH FOR READ WHILE WRITE
- ARITHMETICS
  - * PARALLEL FOR COMMON OPERATIONS ON DIFFERENT DATA
  - * SERIES FOR MORE COMPLEX MACROS
  - * INTERLEAVE FOR HIGHER THRUPUT
  - * PUT IC'S INTO SINGLE PHASE VERSION
- SPECIAL FEATURES
  - * CONFIGURE AU STAGES AS NEEDED FOR HARDWIRED FUNCTIONS

SECTION III
TECHNICAL DISCUSSION

3.1 Problem Statement

A typical avionics processing system is shown in Figure 36 to illustrate the signal processing tasks to be discussed. The signal processor input is a video or IF signal from the receivers. Its output is the processed and reduced data in digital form suitably formatted. This output is transferred to a control computer or display.

The signal processing task is, in brief, to flow a set of data through a sequence of filters one after the other. The objective is to tag the location of a handful of target data points in the vast quantity of noise and/or clutter. A typical filter algorithm is a sequence of operations consisting of memory storage, arithmetic operations (such as addition, subtraction, multiplication) and some data switching.

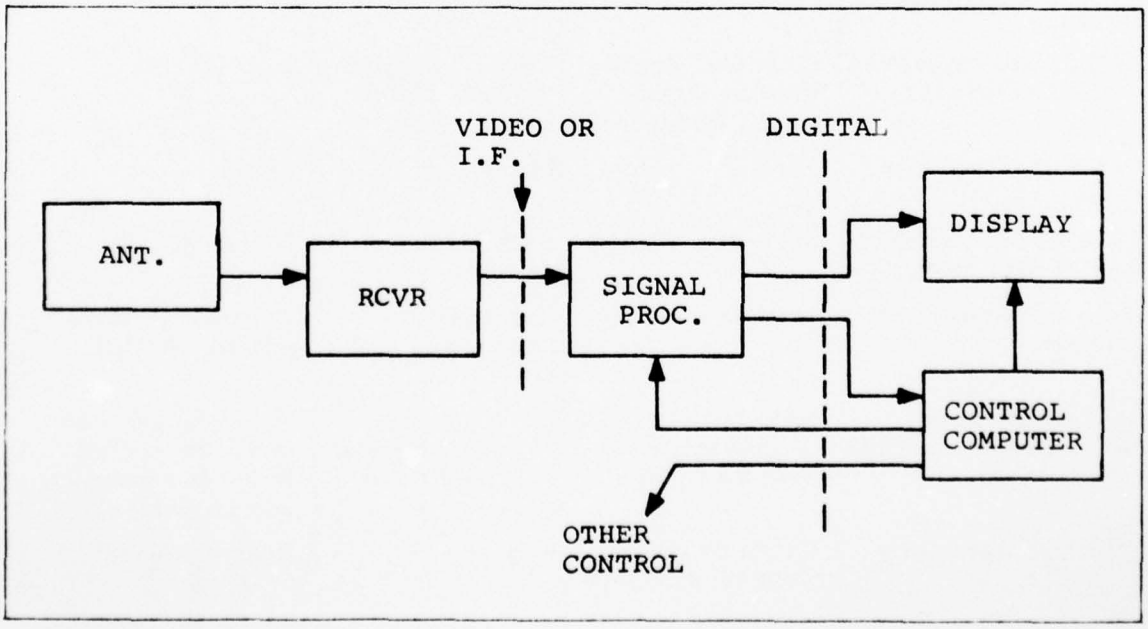


Figure 36- Signal Processor in Typical System

Note also that Figure 36 does not distinguish between those signal processing tasks which are performed by analog hardware and those done digitally. The exact boundary is a function of the performance desired for the price and the state of technology, rather than any fundamental limitations. That boundary is constantly changing in favor of more digital processing. Thus, an all-digital signal processing element is developed, based on consideration of all the required basic signal processing algorithms. Furthermore, this study shall concentrate on programmable approaches to digital signal processing, rather than hardwired (e.g., fixed function) digital processing because of the system advantages outlined in Table 9.

TABLE 9

PROGRAMMABLE DIGITAL SIGNAL PROCESSING ADVANTAGES

Topic	Programmable Digital	Analog	Hardwired Digital
Hardware Utilization on Multimode applications	High	Low	Low
Reliability Improvement Techniques	Graceful degradation; Fractional redundancy	Close Temperature Control; Duplication	Duplication
Number of module types	Few	Many	Many
Ease of accommodation of new modes	Software change	New Module designs	More Modules
Ease of parameter changes	Software changes	Component changes at least	At least system wiring changes
Cost of dynamic range considerations	Automatic scaling is incorporated with no losses	Device precision determines cost and accuracy	Word length and position varies from one function box to another
Efficient operating modes	Batch or real-time or delayed	Real-time	Real-time or delayed

With a general purpose (GP) computer, data points are fed into the main memory either once before calculations or continually on an interrupt basis. A single limited-capability arithmetic section performs all of the required transformations under program control. Each arithmetic calculation (i.e., add or subtract) requires a separate instruction. A sequence of instructions for a given transformation forms a subroutine, and a particular sequence of subroutines corresponds to one processing mode. No specialized hardware design is required.

Use of GP architecture does require, however, that sufficient computation time be available. Data sets must arrive with enough time between successive sets to perform all the required calculations. Alternatively, the average time between arrival of data points on an interrupt scheme must be enough to do an appreciable amount of computation. Problems occur if data arrives at higher rates. A typical GP computer architecture becomes limited on any or all of the following:

- The input/output interrupts reduce available time for real computation
- The time to set up each calculation and do bookkeeping further reduces actual computation time
- The quantity of arithmetic operations required may just exceed the maximum possible capability of the arithmetic hardware
- Parallel processing adds arithmetic capability, but also adds much overhead for coordination and system allocation

Signal processing thruput requirements are higher than those obtainable by GP approaches. Using today's LSI technology, a 16 bit minicomputer with overlaped operation can give add times of 0.25 μsec and multiply times of 5 μsec for a thruput of 700 to 1200 KOPS (thousands of mixed operations per second). Speed-up appendages such as a hardware multiplier and shift barrel can

further extend this GP KOP rate two or three times more. However, thruputs at least an order of magnitude higher require exploiting the restrictive nature of signal processing tasks, such as the regular repetition of identifiable macro arithmetic operations.

3.2 Key Principles and Techniques

The design of the micro signal processor elements (μSP) will be derived by combining top-down analysis of avionic processing requirements with probable future technology directions, and relevant computer architecture concepts. Our experience on similar efforts indicates that a workable degree of linear separation exists between these investigations. Each such investigation adds restrictions on the range of viable alternatives open for the design details of the μSP, leaving the outline of the advocated approach.

Some initial constraints can be placed on the μSP by a quick view of signal processor application boundaries over ground, airborne and missile applications. Table 10 summarizes the normal signal processing categories seen at Raytheon for recent real and proposed applications. Thruput, as measured in terms of real multiplies per second, varies from 0.3 MHz to 60 MHz rates. Storage needs vary from a few thousand bits to a few million bits. Consequently, signal processor sizes vary from thirty chips to ten-thousand chips, with power consumption ranging accordingly, Satisfying this three orders of magnitude variation cannot be done effectively with one design. Hence, we shall eliminate the applications on the large end as unsuitable for defining a μSP. Also the smallest missile tasks will not be considered since they end up as extremely specialized implementations (at least today). Remaining are the small ground applications, small to medium airborne applications, and medium to large missile applications. RPV applications also are suitable for this μSP, being equivalent to a large missile in computation requirements.

TABLE 10
APPLICATION CONSTRAINTS

Use	Throughput (Multiplies/sec) (MHz)	Memory (Kilobits)	IC Quantity (K)	Power (W)	Card Area ($in.^2$)	Packaging Notes
Ground	60 6	2,000 200	10 0.7	5,000 200	≤80	Almost any scheme will do
Airborne	20 2	600 30	3 0.2	900 500	≤40	Atr Dimensions, Cooling problems, Minimum cable quantity
Missile	3 0.3	60 3	300 30	100 10	≤20	Battery and cooling problems, special card shapes, serial I/O essential

Micro Signal Processor Application Objectives:

- Ground - Small Sizes
- Airborne - Small to Medium Sizes
- Missile - Medium to Large Sizes

Signal processing involves the high speed manipulation of sensor data to extract the significant information from the background. Processing also depends on the end use of that information.

- Radar signals - are detected against a white noise background by matched filtering, to enhance signal-to-noise (S/N) ratio and remove clutter masking targets
- Image processing - improves image quality by contrast enhancement, edge enhancement, etc, or extracts prominent features from the background to simplify transmission or subsequent calculations like correlation or tracking
- EW - includes sorting of video pulse descriptions by arrival angles and center frequency unitl pulse repetition intervals can be accurately determined
- Communications - includes speech compression and video band width compression in preparation for secure or jam-resistant transmission or compressed data storage.

Our approach to the analytic aspects of designing these programmable signal processor thus proceed top-down as follows:

Mission - Analysis of system modes and their constituent algorithms bounds thruput, storage, and algorithm variety needs

Function - Analysis of the required algorithms and alternative computation approaches bounds ratios of calculation components, word formats, interconnections

Environment and Technology - Analysis of a particular application environment and a technology snapshot constraints logic type, packaging and many processor features

Arthitecture - From analysis of competitive architectures, firmware tradeoffs, and past experience, modularity directions are defined and a design detailed

Candidate Fit - Trying the postulated design against the evolving system modes tells the size, thruput, ease of use and other measures of design cost/effectivity.

Note that mission analysis is listed before function analysis. Some basic function analysis can occur independent of the mission analysis. However, determination of arithmetic thruput rates, ratios, and accuracy depends on first defining example missions to ensure algorithm exhaustiveness and to explore alternative orderings and computation schemes. Out experience is that there is a factor of two in performance to be obtained by careful task analysis and maniputation to exploit the programmability of digital SPs. Hence performance analysis started under this study at the same time as function analysis.

3.3 Performance Analysis

A signal processor has several levels of programmability. At the highest level are the modes of a system mission which enhance detectability of different kinds of targets in various environments. These are composed of processing algorithms such as described in the preceding subsection. The latter are, in turn formed by a number of passes through the arithmetic unit, with each pass being a macro instruction execution. Macros are created by combining the micro command bit fields of the adders, multipliers, scalers, working registers, and routing.

In this subsection several different signal processing missions are examined in general and in detail. Examples include: synthetic aperture strip map, A/A search and track, fast and slow ground moving target indication, RF signal sorting and classification, and voice coding.

3.3.1 Strip Map - General

In strip mapping, a large area map is synthesized from a set of smaller maps, each of which has a relatively small number or range cells. The system flow diagram is shown in Figure 37. Complex data from A/D converters first goes through a Barker code pulse compression and then motion compensation. Phase shifting of the data with complex multiplications removes the frequency off-set due to antenna squint and frequency changes due to geometric distance from map center. Next, a weighted sum of several PRF samples, formed for each mapped range interval by a low-pass FIR figter, is buffered and corner turned to gather all samples from the same range cell on the ground.

Range data is then spectrum analyzed, with weighting applied before the FFT to reduce $\frac{\sin x}{x}$ sidelobes. After FFT calculations, doppler cells at the ends of the spectrum are dropped. Magnitude and integration of the data yields a map for display.

3.3.2 A/A Search and Track - General

The A/A search and track system chosen uses low PRF waveforms when looking above the horizon and high PRF waveforms when looking below the horizon. In both modes, the waveforms are processed coherently. Combinations of PRF variation and multiple looks provide for resolution of range and doppler ambiguities. We will explain a high PRF operation with eight range gates. Track waveforms are the same as the search waveforms but with two of the range cells filled by data from the monopulse difference channels.

The flow chart of the system is shown in Figure 38. Pulse compression has not been included in this system. First, correcting the signal for I/Q unbalance in the quadrature demodulator permits discrimination of the target from images in the doppler region representing large amounts of ground clutter. Correction factors are determined by pilot pulse measurements.

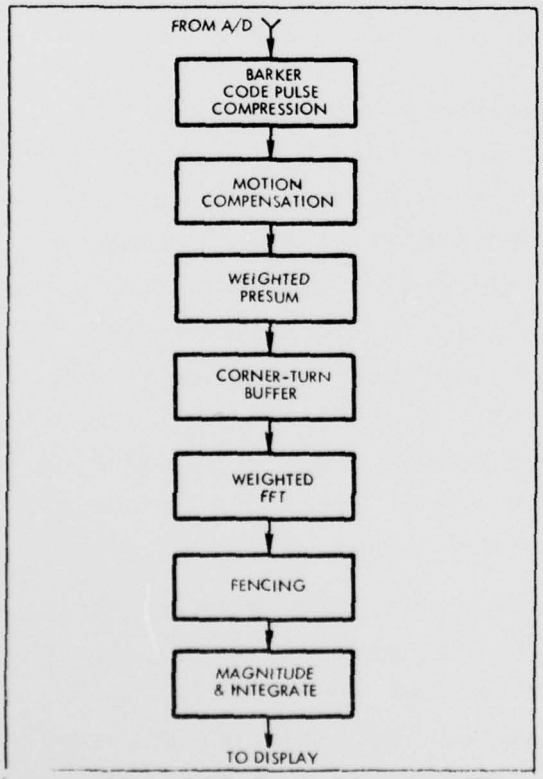


Figure 37 - SAR Strip Mapping Flow

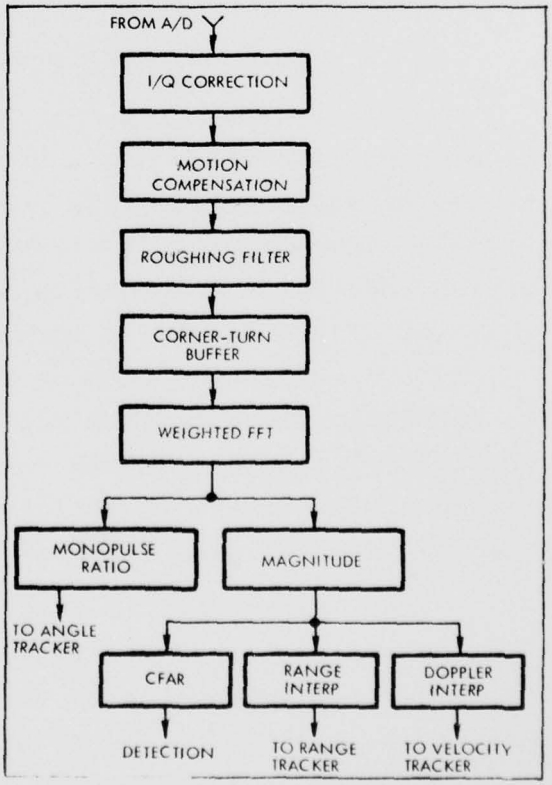


Figure 38 - A/A Search and Track Flow

Then motion compensation by phase shifting the signal occurs, to position the ground clutter for removal by an FIR or an IIR roughing filter. The data is then buffered until all points have been acquired for a spectrum analysis. The spectrum analysis uses time weighting at the FFT input to reduce the $\frac{\sin x}{x}$ sidelobes.

Subsequent operations depend on the operating mode. In the search mode, the complex FFT outputs are magnituded, followed by CFAR detection. In the track mode, the antenna pointing error is computed from the outputs of the sum and difference channels at the predicted target range and doppler. Target range and velocity interpolations are also made.

3.3.3 Ground Moving Target Indication - General

GMTI systems are concerned with fast and slow moving targets. We differentiate between these two by noting that an area on the ground will be illuminated by the main beam of the radar. The doppler shift of fast targets only competes against sidelobe returns of fixed reflectors and are easily detected and tracked with roughing filtering and spectrum analysis. Slow targets present the more difficult resolution problem because their doppler shift competes with the mainlobe returns of fixed targets due to aircraft velocity and pointing angles. A technique under development at Raytheon uses monopulse sum and difference ratios for all instrumented range/doppler cells. In cells containing a slow target and some ground clutter the real and/or the imaginary part of the monopulse ratio will fall outside its expected value, allowing detection. In Figure 39 the sum and difference signals flow through identical processing initially. Motion compensation corrects for nonuniform aircraft velocities and centers the main doppler beam on zero frequency. A roughing filter is applied to each range cell in

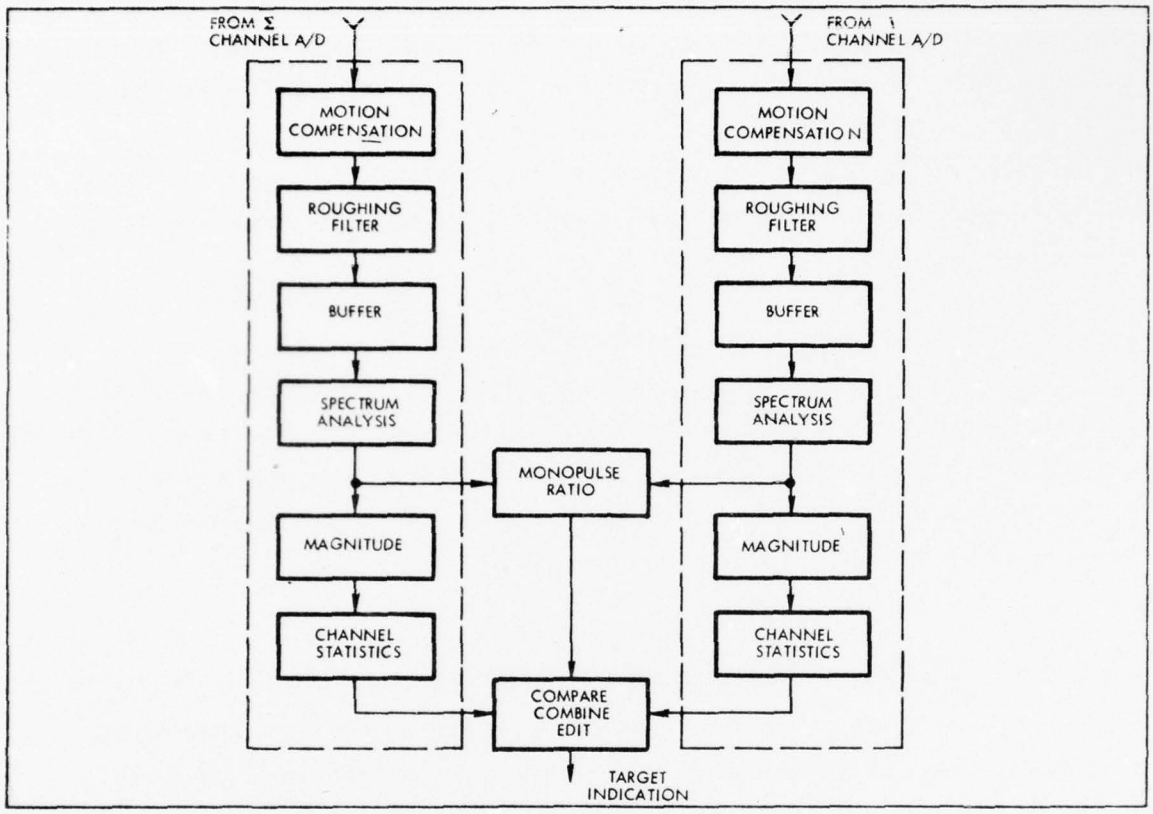


Figure 39 - A Slow COMTF System

the interval where the difference channel is well behaved. After sampling rate reduction, data is buffered, then spectrum analyzed with a weighted FFT. Next, a complex monopulse ratio is formed for each of the instrumented range doppler cells. Then for each doppler cell of the sum and difference channel, an averaging over range is done to establish ground clutter statistical measures. These statistical measures set thresholds against which the complex monopulse ratio will be compared to obtain target indications.

3.3.4 Radar Air to Ground Processing - Detail

A representative set of processing requirements were postulated for an advanced air-to-ground radar system, in order to detail the algorithms required and typical processing

AD-A047 997 RAYTHEON CO BEDFORD MASS MISSILE SYSTEMS DIV F/G 17/9
HIGH SPEED MICRO SIGNAL PROCESSOR STUDY.(U)
SEP 77 G N SHAPIRO F33615-76-C-1339
UNCLASSIFIED AFAL-TR-77-52 NL

2 OF 3
ADA047997

loads. Two major modes were considered;

1) Strip mapping and ground moving target detection (GMTD)

2) Spotlight, ground moving target track (GMTT), and missile retransmission processing (RETRAN).

These are essentially search/acquisition, and target track/missile guidance. For the STRIP MAP and GMTD mode, basic processing uses a presumming filter for bandwidth reduction while the second mode uses FFT roughing filters. Also mode 1 uses complementary Golay code pulse compression with side lobe improvement by addition of sequential returns, while mode 2 uses Golay code pulse compression by replica convolution only.

For mode 1, the flow diagram is presented in Figure 40. An analysis of the computation requirements of each of the constituent functions is presented in Table 11. Clearly, the processing up through motion compensation dominates the total, with Golay pulse compression dominating most of the loading. For example, with typical PRF rates, the Golay code processing is up in the area of several hundred million adds per second. This is beyond the range where the μSP even in netted systems is appropriate, leading to the recommendation that this front-end task be considered for specialized sub-nanosecond logic implementation. Conversely, the processing after the weighted presumming is easily within the postulated capability of only one μSP.

The flow chart and functional analysis for SPOTLIGHT, GMTT and RETRAN mode are given in Figure 41 and Table 12 respectively. It is assumed that missile returns for sum and difference channels are frequency multiplexed together. FFT roughing is used to de-multiplex the F_1 spectrum. Again the major computation loading is around the Golay code pulse compression. The FFT roughing filter however, now represents a significant computation load, say 5 to 10 μSP's worth.

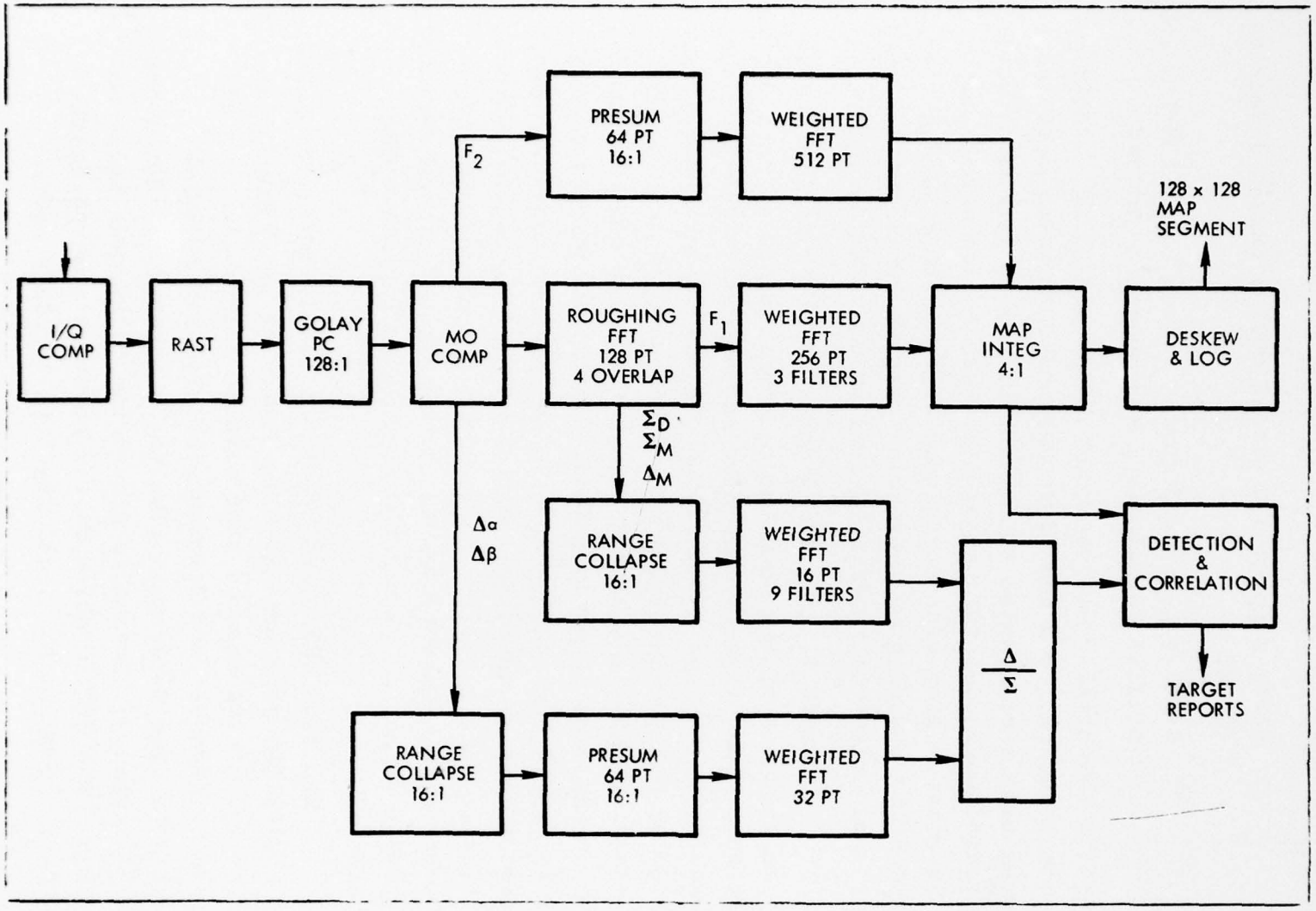


Figure 40 - Flow Chart Spotlight GMTT and Retran

TABLE 11

FUNCTIONAL ANALYSIS SPOTLIGHT MAP, GMTT & RETRAN

SIGNAL PROCESSING FUNCTION	COMPUTATION REQUIREMENTS PER COMPLEX CHANNEL		COMPUTATION RATE	NOTES
	REAL ADDS	REAL MULT		
I AND Q COMP	4	2	PRF x (N + R)	N = GOLAY CODE LENGTH
RAST	1	2	PRF x (N + R)	R = RANGE SAMPLES = FFT POINTS
GOLAY PULSE COMP	4	-	PRF x (N + R) x LOG N	L = FFT ROUGHING FILTER OVERLAP
MOCOMP	2	4	PRF x R	G = ROUGHING FILTER COMPRESSION RATIO
FFT ROUGHING FILTER	6	4	$\frac{PRF}{2}$ LOG x R x L	F = NUMBER OF ROUGHING FILTER OUTPUTS BEING PROCESSED
FFT MAP/TARGET FILTER	6	4	$\frac{PRF}{2} \times LOG \times R \times \frac{L}{G} \times F$	K = RANGE COLLAPSE RATIO
MAP/INTEGRATION	3	-	R	D = DESIGNATED NUMBER OF TARGET RANGE AND AZIMUTH CELLS
RANGE COLLAPSE	2	-	$PRF \times \frac{L}{G} \times R \times F$	P = PRESUM WINDOW
$\frac{\Delta}{\Sigma}$ COMPLEX DOUBLE PRECISION	22	56	$PRF \times \frac{L}{G} \times D$	C = PRESUM COMPRESSION
DESKEW & LOG	3	1	$\frac{R}{2}$	
DETECTION/CORRELATION	20	20	$PRF \times \frac{L}{G} \times D$	
PRESUMMER	2	2	$PRF \times \frac{R}{K} \times \frac{P}{C}$	

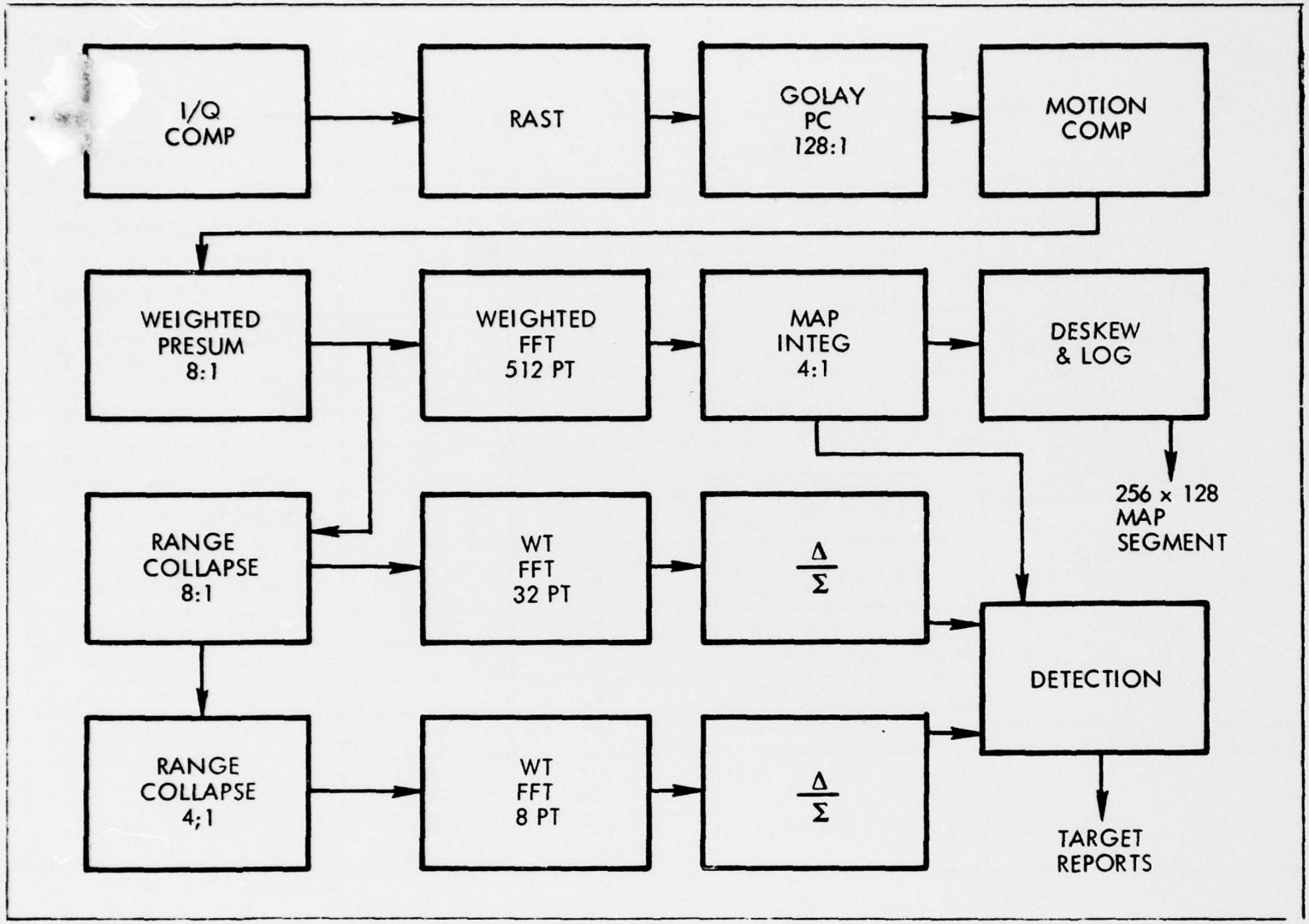


Figure 41- Flow Diagram Strip Map and GMTD

TABLE 12
FUNCTIONAL ANALYSIS SPOTLIGHT MAP, GMTT & RETRAN

SIGNAL PROCESSING FUNCTION	COMPUTATION REQUIREMENTS PER COMPLEX CHANNELS		COMPUTATION RATE	NOTES
	REAL ADDS	REAL MULT		
I & Q COMPENSATION	4	2	PRF x (N + R)	N = GOLAY CODE R = RANGE SAMPLES P = PRESUM WINDOW C = PRESUM COMPRESSION η = FFT FILTERS K = RANGE COLLAPSE RATIO
RAST	1	2	PRF x (N + R)	
GOLAY PULSE COMPRESSION	4	-	PRF x (N + R) x LOG N	
MOTION COMPENSATION	2	4	$\frac{PRF}{2} \times (N + R)$	
WEIGHTED PRESUM	2	2	$\frac{PRF}{2} \times R \times \frac{P}{C}$	
TIME WEIGHTING	-	2	$\frac{PRF}{2} \times R$	
FFT	6	4	$\frac{PRF}{2} \times \frac{1}{C} \times \frac{LOG\eta}{2} \times \frac{R}{K}$	
MAG/INTEG	3	-	ηR	
RANGE COLLAPSE	2	-	$\frac{PRF}{2} \times \frac{1}{C} \times R$	
$\frac{\Delta}{\Sigma}$ COMPLEX DOUBLE PRECISION	22	56	$\frac{PRF}{2} \times \frac{1}{C} \times \frac{R}{K}$	
DESKEW & LOG	3	1	$\frac{\eta R}{2}$	
DETECTION	6	4	$\frac{PRF}{2} \times \frac{1}{C} \times \frac{2R}{K} \times LOG\eta$	

All the remaining processing, including the presum, FFT, and post processing generally can be placed into one μSP.

For both modes a significant amount of bulk memory is required, somewhere in the vicinity of 2 to 8 million bits of memory. When this amount is made of 4K bits per chip, system size is dominated by memory modules. Hence, the use of 16K bits per chips is necessary for memory sizes to be comparable to the advantages obtainable with LSI proccssor sizes. Shift register type memories severely restrict the choice and sequence of processing algorithms. Hence, we emphasize the choice of random access memory, even though this entails dynamic memory elements and the problems of meshing refresh cycles into the high thruput processing scheme.

Trade-offs exist between storage and processing, without degrading display performance. Smaller presumming sizes require larger FFTs with selection of fewer outputs from the FFTs. Without taking sides on this perennial design question, we prefer to push the technological capability to provide significantly greater amounts of both storage and programmable thruput capability in smaller spaces. That is the fundamental thrust of μSP development.

3.3.5 Linear Predictive Processing

3.3.5.1 Introduction

This subsection details some typical processing tasks involved in a communication-type application. Adaptive predictive processing represents a category of computations which can be handled efficiently within a micro signal processor. This category is also representative of higher thruput video communications signal processing, where the primary objective is a drastic reduction in the video bandwidth necessary to represent the sensor information before entering into a jam-resistant transmission process.

A growing interest can be expected in the application of these algorithms to radar target detection and tracking. A drawback is that the processing load increases several times over conventional frequency domain processing with FFT's and CFAR's, but the payoff is greater in the enhanced discrimination of closely space returns. Examples are small targets in the presence of a large target or jammer, and moving target indicators. Figure 42 summarizes these concepts.

Incidently, one of the differences between the various batch type adaptive discrete filtering schemes is in the assumptions made about the data behavior outside the batch interval. For example, the maximum likelihood method when used to operate an autoregressive filter assumes that the data is zero outside the interval. The maximum entropy method makes no assumptions whatsoever. In contrast the standard FFT assumes that data is periodic outside the region given, with a fundamental period equal to the batch size.

In communications, a Vocoder represents one of the more fruitful areas where the Micro Signal Processor could be applied. At least two applications of this type have been documented. Weinsteine [1] * described the use of the Lincoln Labs Fast Digital Processor as a Linear Predictive Vocoder. Goldberg and Arcese [2] showed that adaptive predictive encoding could be done using the Sylvania Programmable Signal Processor.

These Vocoders convert analog speech into a digital representation for transmission on a communications channel. During processing they compress the speech and reduce the bit rate on the channel by a factor of about 10 while maintaining the speech quality.

There are two primary motivations for digitizing the speech signal. First it greatly simplifies and ecomomizes

* Note: References indicated are those found in section 3.3.5.6

BATCH TYPE:	AUTOREGRESSIVE/MAXIMUM ENTROPY FILTERS
	APPLICABLE TO: SPEECH COMPRESSION
	SAR MAPPING
	FREQUENCY TRACKING
	YIELDS IMPROVED FREQUENCY RESOLUTION RELATIVE TO BATCH SIZE
	USED TO LOCATE SPECTRAL COMPONENTS OF COMPLEX SIGNALS
CONTINUOUS:	KALMAN FILTERS
	APPLICABLE TO: NAVIGATION
	STABILIZATION
	TRACKING
	USED TO ESTIMATE PARAMETERS OF SIGNAL CORRUPTED BY NOISE
	ESTIMATED PARAMETERS: FREQUENCY, RANGE, ANGLE, AMPLITUDE

Figure 42- Adaptive Discrete Filtering

any repeaters required in the system. Second it admits the use of encrypting to obtain the advantages of a secure communications channel. In this study, one form of Vocoder is postulated and the processing load imposed by it on the signal processor is evaluated. This system follows the work described by MARKEL [3,4,5,6,7].

3.3.5.2 System Concept

A Vocoder system consists of: a) a voice digitizer that accepts speech and converts it to a compressed digital representation, b) a digital communications channel that carries the digital message from the originator to a receiver, and c) a voice synthesizer that synthesizes a speech signal from the digital representation. These elements are arranged

into a system as shown in Figure 43.

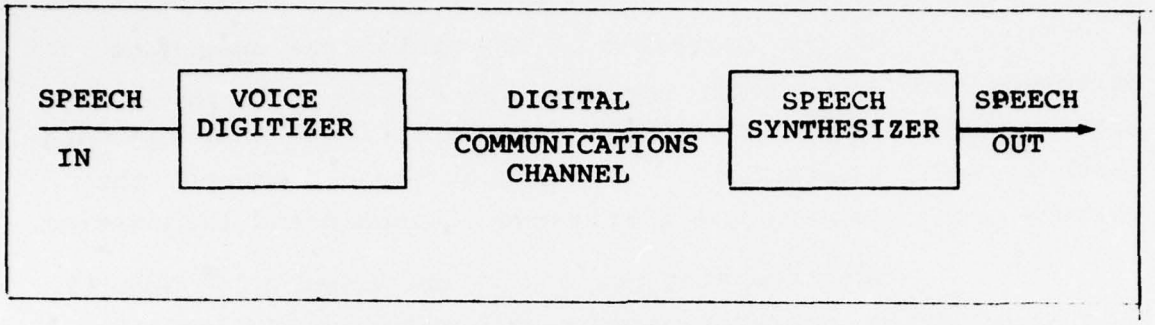


Figure 43 - Vocoder System

The ability to reduce the bit rate required for the speech is based on the model of the vocal mechanism shown in Figure 44.

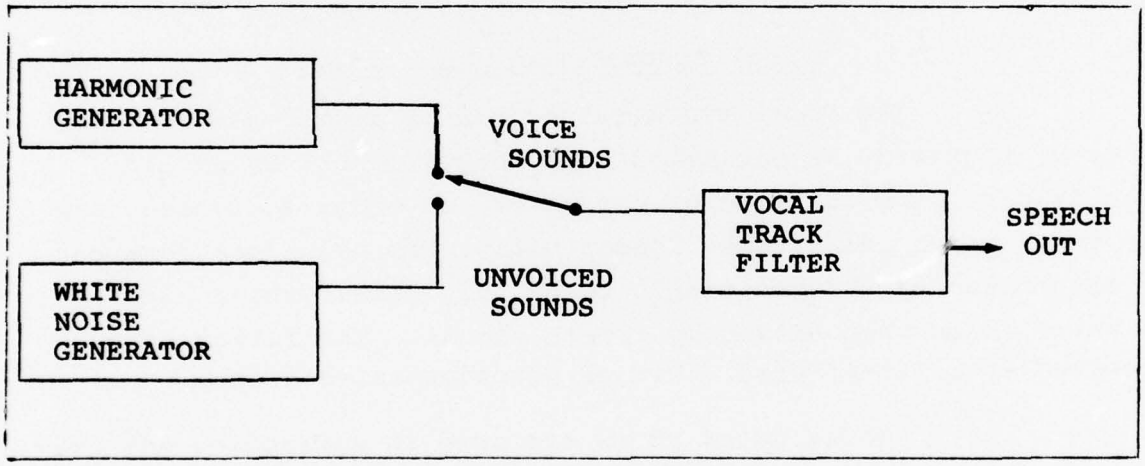


Figure 44- Model of Vocal Mechanism

In this model speech is assumed to be either a harmonic signal whose spectrum is shaped by the vocal track filter for voiced sounds, or white noise that is shaped by the vocal track filter for unvoiced sounds. Although speech contains high frequency signals, the vocal mechanism can only modulate these signals at some low rate.

If for this system we can determine signal amplitude, voiced or unvoiced, pitch if voiced, and about 12 vocal track parameters as a function of time, realistic voice

reproduction can be realized at the receiving station.

In the operation of the system the speech is filtered, sampled, and A/D converted as a first step in operation. The speech samples are then partitioned into segments about 25 milliseconds long. For each of these segments, the various model parameters are extracted, encoded and transmitted.

The receiving station is described in Figure 43. Here, a harmonic generator, a noise generator, and a voiced/unvoiced switch are controlled by the appropriate model parameters. The resulting output is filtered by an all pole filter whose characteristics are determined by the vocal track parameters.

3.3.5.3 Vocal Track Filter Coefficients

The vocal track filter can be represented (to the accuracy required, as demonstrated by experiments) by an all pole filter. The location of the poles, or their equivalent are the vocal track parameters. Conceptually, the pole locations can be determined by synthesizing a zeros only filter whose output is white noise when driven by speech signal. The filter so synthesized is a linear predictive or autoregressive filter.

Three major steps are used in extracting and preparing the vocal track parameters for transmission. First the autocorrelation function for each set of input samples is generated. This calculation can be accomplished using FFT procedures, or by an accumulation of products procedure. For the size of the sets, and the number of autocorrelation coefficients extracted, the latter procedure is slightly more economical. In either case, this computation is the major portion of the computing load.

In the second step, the coefficients of the autoregressive filter are determined. The autoregressive filter

has a transfer function.

$$H(z) = \sum_{i=0}^{M} a_i \, z^{-1} \qquad a_o = 1$$

The set of coefficients $[a_i]$ of this filter are detemmined by an algorithm of the type shown in Figure 47. This algorithm is a simplified version of Robinson's algorithm [8] as developed by MARKEL and GRAY [5].

The third step is to prepare the coefficients for transmission. Fettweis [9] showed that Digital wave filters require fewer digits for the multiplier coefficients than more conventional structured IIR filters. By transforming from the filter coefficients $\{a_i\}$ to wave filter parameters $\{k_m\}$ the amount of data required for transmission can be reduced. An algorithm for making this transformation was described by GRAY and MARKEL [6].

From a signal processor viewpoint, these several algorithms consist primarily of real adds and multiplies with an occasional divide. The algorithms have a regularity that make them amenable for use in a signal processing structure. In the Robinson Algorithm, Markel and Gray [5] showed by simulation that 19 to 22 bits will be required in floating point calculations to insure the stability of the filters.

The wave filters as described by k_m are precisely equivalent to the "PARACOR COEFFICIENT" derived by ITAKURA and SAITO [10]. Therefore, that form of voice processing to determine vocal track parameters is included within the signal processing structure being examined.

3.3.5.4 Other Parameters

The other parameters of the system include: a voiced/unvoiced decision, pitch for voices sounds, and signal level. The general procedure is to process extensively to get a good chance of making a correct choice and then clean up the results using a set of logical decisions. The processing up to this last point is of the type normally done in signal processing. The logical processes will require either a small GP computer, or a look-up table decision network. The processing steps which follow Markel and Gray [7] include the following:

1. Test the zero crossing rate of the input samples at an 8 KHz rate. If less than 2 crossings per millisecond tag the segment as unvoiced.
2. Low pass filter the signal using a 3 pole Chebyshev filter with an 800 Hz cut off frequrncy. Decimate the output sampling rate to 2 KHz to form the test segment.
3. Remove any bias that is present in the test segment.
4. Measure the power level of the segment. Tag segment as silent if a preset threshold is not exceeded.
5. Form the first four auto correlation coefficients of the bias free test segment. Use accumulated products.
6. Use the Robinson Algorithm to obtain a four coefficient auto regressive filter.
7. Filter the bias free test segment using the auto regressing filter.
8. Form the autocorrelation function of the residue from the auto regressive filter. Use FFT procedures.
9. Locate the peak of the auto correlation function. This is a coarse indication of pitch period.
10. Interpolate using the auto correlation peak and the adjacent two samples at six intermediate points.
11. Pick the maximum from the interpolated values as the locations of the pitch period.

12. Test the maximum against a threshold for a voiced/unvoiced decision.
13. Do logical tests to clean up.

3.3.5.5 Processing Load

The total processing requirments thus imposed by the algorithms just described are indicated in Figure 45. Further breakout of the "other" parameter processing is presented in Figure 46. Note that the rates are more than an order of magintude below that achievable by a signal/μSP. Hence, a number or options are possible:

- Multiplex many such channels through one unit
- Squeeze this processing onto a μSP for another task
- Consider using some of these adaptive algorithms for higher thruput sensors, namely radar

More detailed insight into the algorithms is provided by Figures 47 and 48. The former shows the formulae for the autoregressive filter while the latter shows the formulae for the Berg algorithms for maximum entropy calculations.

The digital lattice filter form is part of the vocoder scheme just described. It was invoked because of claims of minimizing coefficient bits. However, further literature search has revealed that the more conventional form of IIR filter, namely cascaded two-pole, two-zero sections, is as good in that property. Hence, the law of simplicity is invoked, and reliance on the conventional form will be assumed.

SYSTEM PARAMETERS:

SAMPLING RATE	8 kHz
FRAME RATE	40/sec
NUMBER OF COEFFICIENTS	12.

VOCAL TRACK PARAMETER PROCESSING:

	MULTIPLIES PER FRAME	ADDS PER FRAME	DIVIDES PER FRAME
AUTOCORRELATION	2400	2400	-
ROBINSON ALGORITHM	180	180	12
LATTICE CONVERSION	72	72	12
TOTAL/FRAME	2652	2652	24
TOTAL/SECOND	105,080	105,080	480

OTHER PARAMETERS

THE OTHER PARAMETERS OF THE SYSTEM INCLUDE: A VOICES/UNVOICED DECISION, PITCH FOR VOICED SOUNDS, AND SIGNAL LEVEL

* TOTAL PROCESSING LOAD:

	MULT/sec	ADDS/sec
VOCAL TRACK	105,000	105,000
OTHERS	131,000	178,000
TOTAL	236,000	283,000

Figure 45 - Linear Predictive Vocoder

PROCESSING STEP	MULTIPLIES PER FRAME		ADDS PER FRAME
1. ZERO CROSSING TEST			200
2. LOW PASS FILTER	600		600
3. BIAS REMOVAL			100
4. POWER LEVEL	50		50
5. FOUR AUTOCORRELATION COEF	200		200
6. ROBINSON ALGORITHM	20		20
7. AUTOREGRESSIVE FILTER	200		200
8. AUTOCORRELATION VIA FFT	1900		3000
9. LOCATE PEAK			50
10. INTERPOLATE	20		20
11. PEAK PICK			20
12. THRESHOLD		MILL	
13. LOGIC		MILL	
TOTAL PER FRAME	3290		4460
TOTAL PER SECOND	131,600		178,400

Figure 46 - Other Parameters Processing Load

INPUT: 1. A set of samples x_n (n=0,1,. . . n-1)

2. The order of the filter M

OUTPUT: The set of filter coefficients $A_{M,\ell}$

A) Compute autocorrelation function of the input data.

$$v_k = v_{-k} = \sum_{n=0}^{N-1-|k|} (x_n \; x_{n+|k|})$$

B) Using the initial conditions

$$\alpha_0 = v_0$$

$$\beta_0 = v_1$$

$$A_{0,0} = 1$$

Solve recursively for $A_{m,\ell}$ until m = M using

$$k_m = - \beta_m/\alpha_m$$

$$A_{m+1,\ell} = \begin{cases} A_{m,\ell} = 1 & ; \quad \ell = 0 \\ A_{m,\ell} + k_m (A_{m,m+1-\ell}) & ; \quad \ell = 1,2,\ldots, m \\ k_m & ; \quad \ell = m+1 \end{cases}$$

$$\alpha_{m+1} = \alpha_m + k_m(\beta_m)$$

$$\beta_{m+1} = \sum_{\ell=0}^{m} A_{m,\ell} \; v_{m+1-\ell}$$

REFERENCE: (1) John D. Markel and A.H. Gray Jr. "On Autocorrelation as Applied to Speech" IEEE Trans. Audio and Electro Acoustics Vol, AU-Z1, No 2 April 1973 pp 69-79.

Figure 47 - Algorithms for Autoregressive Filter

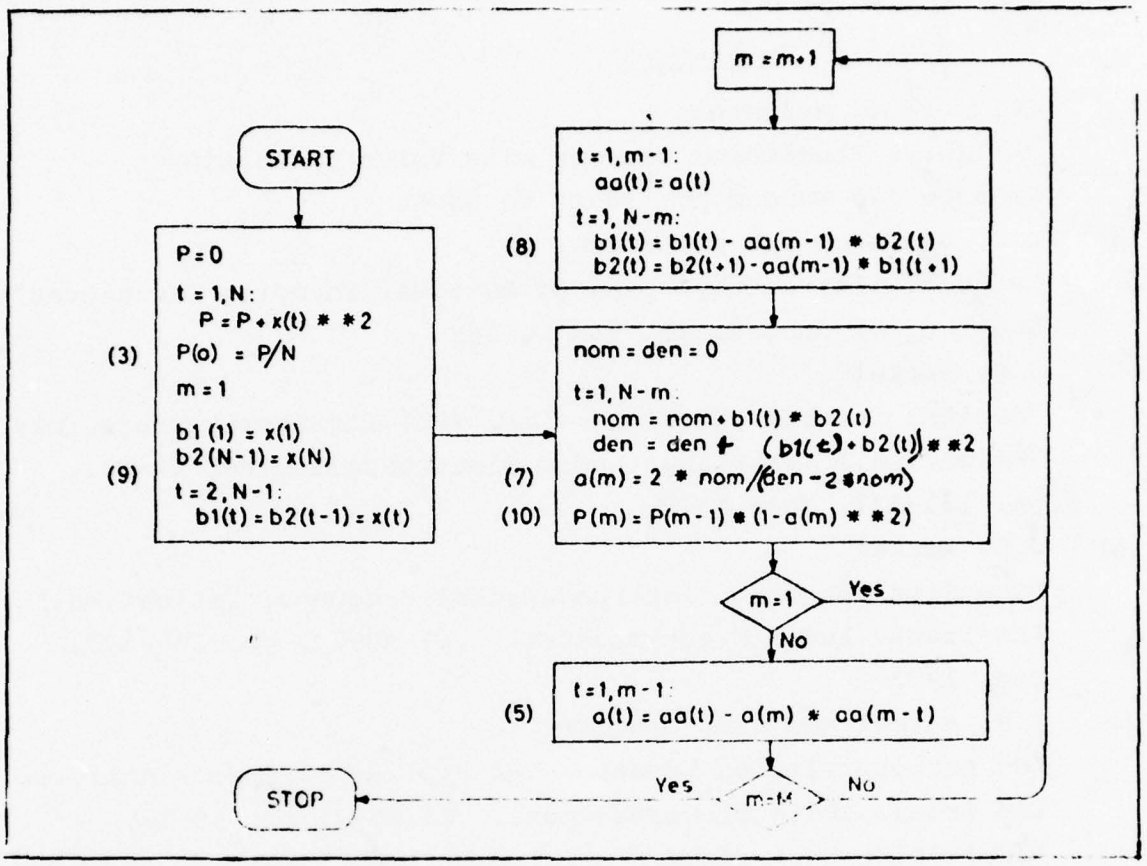


Figure 48 - Calculations For Maximum Entropy Filter*

---

* Reference: N. Andersen, "On The Calculation of Filter Coefficients For Maximum Entropy Spectral Analysis," (2) Geophysics, Vol. 39-No.1 (Feb. 1974), P. 69-72.

3.3.5.6 References

(3) Clifford J. Weinstein
"A Linear Prediction Vocoder with Voice Excitation"
Eastcon '75 Record pp. 30-A, to 30-G.

(4) A.J. Goldberg and A. Arcese
"A Quantitative Comparison of Residual Encoding Techniques"
Eastcon '75 Record, pp. 28A to 28F

(5) J.D. Markel
"Digital Inverse Filtering a New Tool for Format Trajectory Estimation," IEE Trans. Audio Electroacout., Vol AU-20, pp. 129-137, June 1972

(6) J.D. Markel
"The Sift Algorithm for Fundamental Frequency Estimation," IEE Trans. Audio Electroacoust., Vol AU-20, pp. 367-377, Dec. 1972

(7) J.P. Markel and A.H. Gray Jr.
"On Autocorrelation Equations as applied to Speech Analysis", IEE Trans. Audio Electroacoust., Vol AU-21 pp. 69-79, April 1973.

(8) A.H. Gray Jr. and John P. Markel
"Digital Lattice and Ladder Filter Synthesis,"
IEEE Trans, Audio Electroacoust., Vol AU-21, pp. 491-500; December 1973

(9) J.D. Markel and A.H Gran Jr.
"A Linear Predictive Vocoder Simulation Based Upon the Autocorrelation Method,"
IEEE Trans. Acoust, Speech, Signal Processor, Vol ASSP-21, pp. 124-134, April 1974

(10) E.A. Robinson
Statistical Communications and Detection with Special Reference to Digital Data Processing of Radar and Seismic Signals
New York: AAFNER, 1967, pp. 274-279

(11) Alfred Fettweis
"Pseudo Passivity, Sensitivity, and Stability of Wave Digital Filters,"
IEEE Trans, Circuit Theory, Vol CT-19, pp. 668-673, November 1972

(12) Papers B6-10, Cr, D7, H10, and L4 in conf. rec. 1972 IEEE Conf, Speech Commun. and Processing

3.3.6 RF Signal Sorting and Classification

This last example is from ECM systems, and explores the need for high speed signal sorting rather than number crunching. The purpose is to determine the PRI and PRI type (constant, staggered,...) of hundreds of simultaneous emitters. The system block diagram is just Figure 3-1 without a transmitter. An omnidirectional antenna feeds a receiver where the pulse frequency (F), angle of arrival (AOA), and time of arrival (TOA) are assembled into a pusle descriptive word (PDW). The signal processor screens PDWs into the following emitter types based on F, AOA, and previous PDWs:

- New emitters are stored with an activity count of 1
- Old ones have their activity count incremented and new TOA saved
- N'th time ones have their F, AOA and all TOAs go to the data processor
- Beyond N times, the PDW is ignored

The data processor then does the actual PRI and PRF type calculations at msec rates compared with signal processor inputs at μsec rates.

A signal sorter implementation using content addressable memory is very powerful but not recommended for this study. These special components lack the intensive commercial investment

activity of RAMs.

The approach initially preferred here is based on hash coding of the F and TOA part of a PDW to address a RAM. The extra steps required for multiple hits of the same address mapping are minimized by intelligent choice of mapping and by keeping memory loading below half capacity.

This second approach is preferred because it uses conventional RAMs and micro processor CPU slices. However, the computer which does such sorting is still fundamentally different than a number cruncher. Fast, efficient handling of this tag mapping and matching at first appeared to require a unique building block. If this were true, the remaining elements in this application could still be common to those filtering type missions. The approach finally developed emphasizes the direct list processing capability for the μSP. A feature is included in the address generation mechanism to allow a data word to later serve as an address pointer to another data word, etc., Such list structures can branch or terminate upon meeting appropoiate arithmetic conditions.

This ability to work with pointers to data subsets has applicability far beyond the ECM problem. Signal post processing is expected to head in this direction as more experience is gained by analyiis with the unique abilities of digital signal processes. Finally, the ability to expand and compress data sets is one of the key useful functions to emerge from recent examinations of the strengths of the first generation of vector computers such as CDC STAR and TI ASC.

3.3.7 Processing Modes

The μSP is expected to be capable of performing the functions listed in Table 13. This table is based on the previous analysis, as well as summarizing Raytheon's experience

over the past six years of applying programmable signal processors. The degree of efficiency and proficiency at any given function can vary, depending upon the frequency of execution required as well as the particular configuration of μSP elements chosen to satisfy that particular application. Further breakdown of these algorithms into more basic elements occurs under the next section, functional analysis.

TABLE 13

EXPECTED μSP ALGORITHMS

- CORNER TURNING OF DATA
- FFT, FFT^{-1}
- HETERODYNE
- BINARY PHASE CODE CORRELATION
- FIR - CONTINUOUS
- FIR - SUM AND DUMP
- IIR
- MAGNITUDE AND INTEGRATE
- AUTOREGRESSIVE/MAXIMUM ENTROPY FILTER
- CFAR AND THRESHOLDING
- DATA SORTING
- PSEUDO RANDOM NUMBER GENERATION
- KALMAN FILTERING

3.4 Functional Analysis

Missions can be broken down into their constituent algorithms, such as done in Figure 49 for radar missions. As we have seen in the previous sections, similar constituent algorithms can be found in the other missions analyzed. This section examines those fundamental algorithms in more detail in order to derive design restrictions.

	A/D CONVERTER	PULSE COMPRESS	I/Q COMPENSATION	FOCUS/MOTION COMPENSATION	RANGE SAMPLE INTEGRATE	ROUGHING FILTER	SPECTRUM ANALYSIS	MAGNITUDE	INTEGRATE	SCALE/NORMALIZE	BLANK/FENCE	MONOPULSE ANGLE	ADAPTIVE THRESHOLD/COMPARE	THRESHOLD GENERATOR/CFAR	DATA COMPRESS	FORMAT DISPLAY DATA	GROUP AND INTERPOLATE REPORTS	REPORT TO COMPUTER	CORRELATOR
PPI MAP	X	X			X				X	X				X	X	X			
DBS MAP	X	X	X	X	X	X	X	X	X	X					X	X			
FIXED SQUINT MAP	X	X	X	X	X	X	X	X	X	X					X	X			
TELESCOPE MAP	X	X	X	X	X	X	X	X	X	X		X			X	X	X		
DOPPLER VELOCITY SENSE	X		X		X		X	X		X	X	X			X			X	
MAP MATCH	X	X	X	X		X	X	X	X	X								X	X
A/A SEARCH	X		X				X	X			X		X	X			X	X	
A/A TRACK	X		X				X	X				X	X	X				X	
GMTI SLOW	X	X	X		X	X	X	X	X	X	X		X	X			X	X	
GMTI FAST	X	X	X		X	X	X	X	X	X	X		X	X			X	X	
GMTT SLOW	X	X	X		X	X	X	X	X	X		X	X	X			X	X	
GMTT FAST	X	X	X		X	X	X	X	X	X		X	X	X			X	X	
TF/TA	X	X			X				X	X			X	X	X	X	X	X	

Figure 49 - Functions Composing Missions

3.4.1 A/D Converter

Some key interface design parameters emerge from the need to convert analog sensor data into digital words. Conversion rates below 1MHZ allow one small package to produce an 8 bit word, even when all those bits are not needed. Any extra bits produced can reduce the need for analog AGC, through adoption of some degree of block scaling up to true floating points.

Conversion speeds in the MHz range are sustainable with even a μSP if sufficient buffering is provided. A low-duty cycle can then be the processing time limitation rather than the peak input rate. Alternately, if the information bandwidth is small enough, buffering can make batch-mode demodulation feasible on even high duty cycle inputs.

The μSP should be capable of easily processing complex or real data. Complex data (in-phase and quadrature) conversion is normally used on radar systems because this reduces bandwidth requirements. Image processing uses real data initially. With the exception of Fourier transforms, the processing tasks that follow in most cases which use complex data are equivalent to processing with two real-channel processors. Hence, our recommendation that the μSP be equally efficient with complex or dual real formats.

The connections between the converter and the μSP are thus possible at any of three places:

- an intermediary buffer memory
- directly into the minimal configuration
- via the GP I/O bus

Provisions to accomodate data introduction and removal at any or all of those three positions have been made in the proposed design. The first connection, via buffering, allows load-while-process operations of the simplest kind. The second

minimizes chips for cases where only small amounts of input buffering are needed. The third connection allows the GP computer to share the analog interface elements at a cost in programming complexity.

3.4.2 Pulse Compression

Commonly used forms of pulse compression include Barker code and linear FM, with frequency and/or time weighting controlling sidelobe levels. The algorithm used depends on both kernel size and signal processor flexibility. For example, linear chirp convolvers are not as efficient as Fourier techniques for sizes over 50 point or so. Binary codes require fewest adds/subtracts when processor data addressing can be flexible. The memory capacity may exceed twice the kernal size, if continuous compression through overlapped processing is desired.

The efficiency of a μSP in terms of the ratio or overhead logic to needed multiplications, additions, and storage, must be high. Otherwise, a special-purpose pulse compression unit will pop up in many applications and thereby reduce the volume of μSPs produced. Furthermore, if the relative advantage of a pulse compressor box is too great, other calculations, such as map correlation and arbitrary filtering will be converted to exploit the pulse compressor.

3.4.3 Range Sample Compression

Combining several range samples at the start of processing drastically reduces the computation load. This is possible when the range resolution of the system exceeds the resolution required of the current mode. Methods vary from simple integration to complicated adaptive moving target indicator (AMTI) filters. Figure 50 shows a three point MTI, with its need for 4 real multiplies and 4 read subtractions per input point.

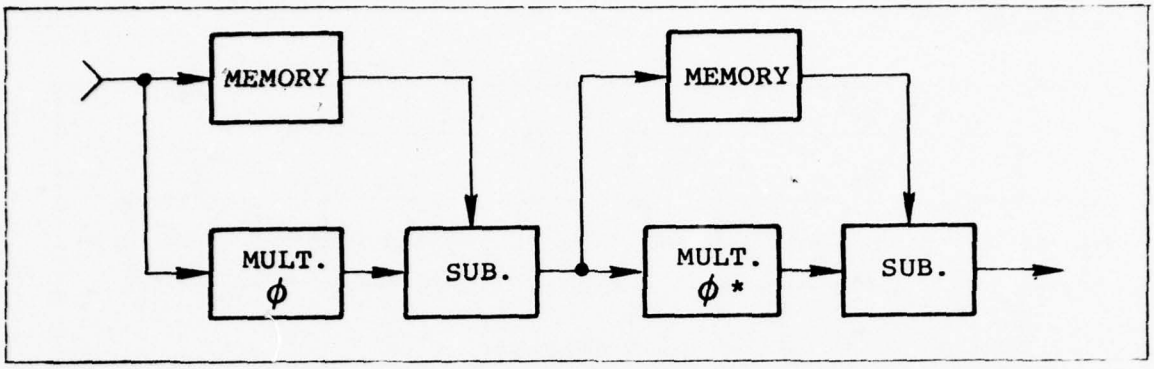


Figure 50 - MTI

3.4.4 Roughing Filters

Roughing filters reduce signal bandwidths, and thus reduce the required sampling rate. In the past typical implementation has been by recursive filters such as combinations of the two pole, two zero module shown in Figure 51. Today the finite impulse response (FIR) filter is replacing such IIR types of filters by offering more performance and/or less computation. Particularly with a μSP, the number of computations required for a FIR filter such as shown in Figure 52 can be manipulated depending on kernel length, bandwidth reduction, and configuration cleverness.

3.4.5 Spectral Analysis

The FFT for spectral analysis has emerged as a signal processing fundamental. Actually the FFT or FFT^{-1} are just a collection of FIR filters whose common calculation is very efficient. Figure 53 shows a base two configuration, or "butterfly" for a FFT stage. For a μSP, higher bases add too much complexity per stage to compensate for their overall computation reduction. Multiple arithmetic units can increase the FFT

thruput, provided memory modularity constraints don't add too much overhead.

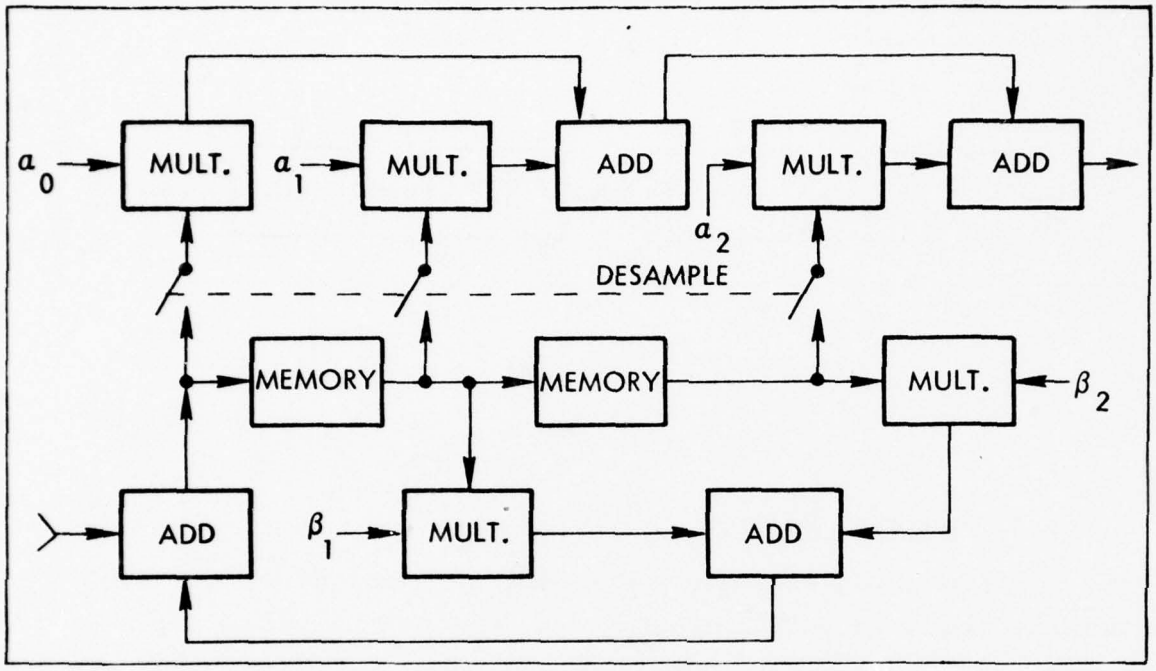


Figure 51 - Two Pole, Two Zero Module for Generalized Recursive Filter

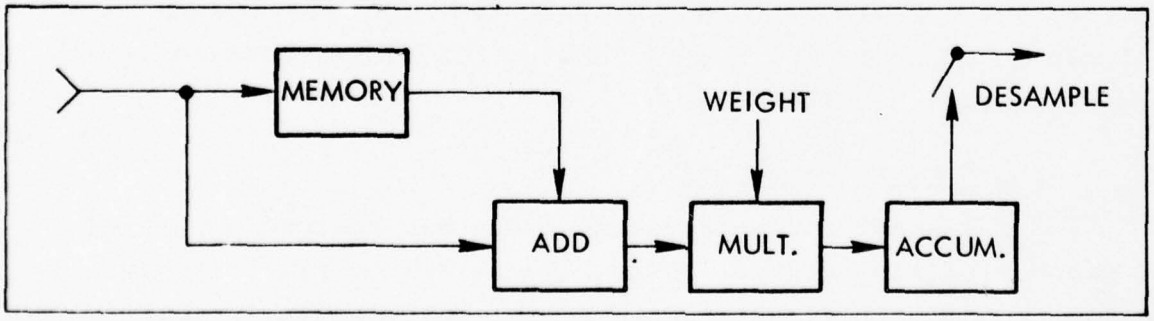


Figure 52 - FIR Filter

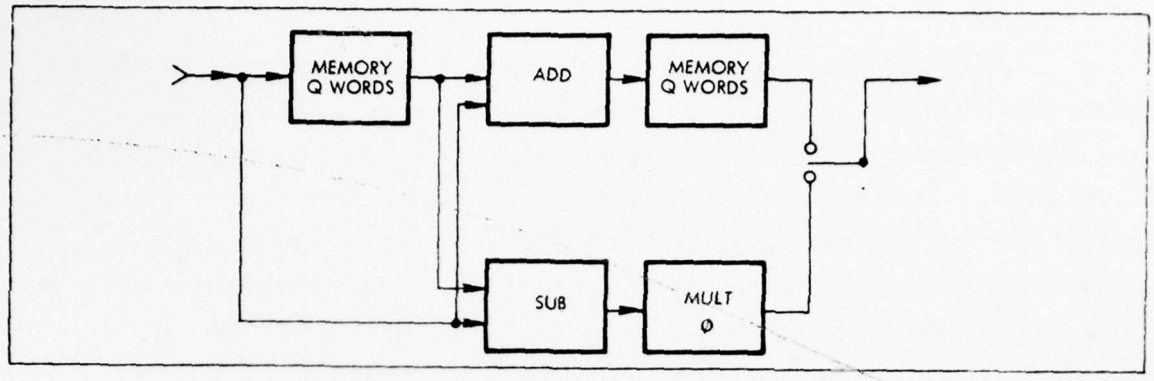


Figure 53 - FFT

The FFT calculation lends itself to many permutations, such as multiplier position before, rather than after, the add-subtracts, and coefficient rearrangement. A possible problem is the bit-reversed order of the FFT output which has been solved by having the control for the μSP capable of reordering up to 8 stages of FFT on one step simultaneous with the next processing operation.

Another possible problem is efficient use of the FFT on real input data since the FFT is defined only for complex data. Here Raytheon had developed a method of computing a real N point spectrum with close correspondence to the flow for computing a complex N/2 point spectrum. Such techniques affect the AU control design flexibility.

3.4.6 Walsh/Hadamard Transform

The Fast Walsh/Hadamard Transform is similar to the Fast Fourier Transform. The FWT is more economical because all of its multiplications are by ± 1. There is a question as to how much importance should be attached to the FWT in the Micro-Signal Processor Study. Based on the discussion below, the FWT is of limited importance and does not influence the μSP features. A FWT can always be done as an FFT with multiplications set to ± 1 These remarks are valid regardless of whether the FWT is used for spectral analysis, convolution, or communication.

Spectral analysis has been used on radar systems to (a) improve the S/N ratio of narrow band targets immersed in noise (eg, pulsed doppler radar), and (b) resolve targets in SAR applications. The major deterent to the use of the FWT in these applications is the problem of interpreting the difference between the Fourier frequency response and the Walsh sequency response. Particularly confusing is the fact that a change in position of a signal has almost no effect on the frequency response while it significantly changes the sequency response.

The class of functions over which the FWT can be used for fast convolution is more restricted than the class of functions for which the FFT can be used. For radars the restricted set of usable functions is much too restricted. The application of the FWT to communication problem has appeared frequently in the literature (See bibliography in N.E. Blackman, Sinusoids versus Walsh Functions, Proc. IEEE Vol 62 pp. 346-354, 1974 [13]). Despite these attempts to use the FWT in communications, it has not been applied to any extent in working systems the way the FFT has.

In communications systems, there has been a continual quest for methods that reduce the bandwidth of transmitted information. This occurs because the number of transmission channels is limited as in radio, or expensive as in telephony. By increasing the cost of terminal equipment, it is hoped to reduce the bandwidth and obtain better utilization of the channel.

Of immediate concern, is a comparison of the FWT as against the FFT in reducing signal bandwidth while giving consideration of the relative cost of these two approaches. Interpreting Blackman's results (FEEC. IEEE Volume 62, p.347, 1974), under the most favorable conditions and for the same accuracy of transmission, the FWT requires 150% of the bandwidth required by the FFT. This comparison is less favorable to the FWT when the most favorable conditions are not present.

At the same time this occurs, of one should delete all the multiples from a generalized signal processing system he would save only about 20% of the hardware complexity. Such a trade off is not attractive enough to bring about extensive application of the FWT in communications systems.

An alternative to FFT for spectral analysis is the Maximum Entropy Filter. The latter is noted for its ability to discriminate between two closely spaced targets. Its computational requirements for N Real input points and M targets (eg. ^{M}iterations) is about 4NM Real multiplies, 4NM Real adds and M divides. For M 2 and N 2^{2M}, the FFT approach requires fewer calculations.

3.4.7 Magnitude and Integration

Magnituding removes the phase angle information from complex data. Integration builds up the signal strength by averaging over several turns. Figure 54 illustrates the arithmetics involved based on a magnitude approximation using the larger and smaller of the real and imaginary components. For greater accuracy, particularly with coordinate conversions, a simple trig function technique can give better than 1 percent accuracy. Such a concept has been refined into a novel scheme, with patent being applied for, to do magnituding in the μSP by angle rotation. Provisions exist for iteration to even higher accuracy.

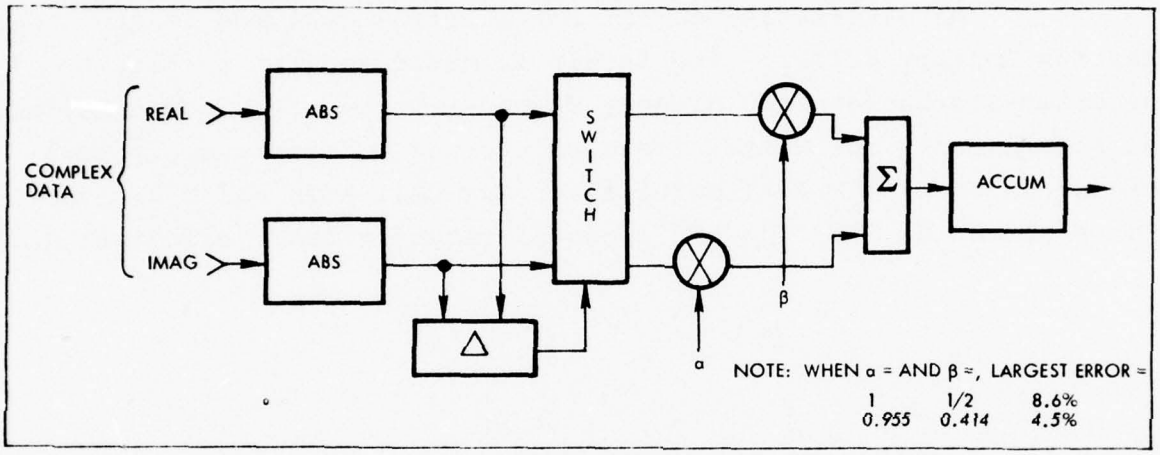


Figure 54 - Magnitude and Integrate

3.4.8 Adaptive Threshold Generation (CFAR)

Sliding window averages of CFARs (Constant False Alarm Rate) compute the local noise background for setting target detection thresholds. The probability of an incorrect judgment, or false alarm rate, can thus be kept fairly constant over a large search area. Figure 55 shows the arithmetic computations

involved, with typical window sizes of 8, 16, 32 and 64 points. A target exceeding a relative threshold is then tagged with a range and doppler identifier.

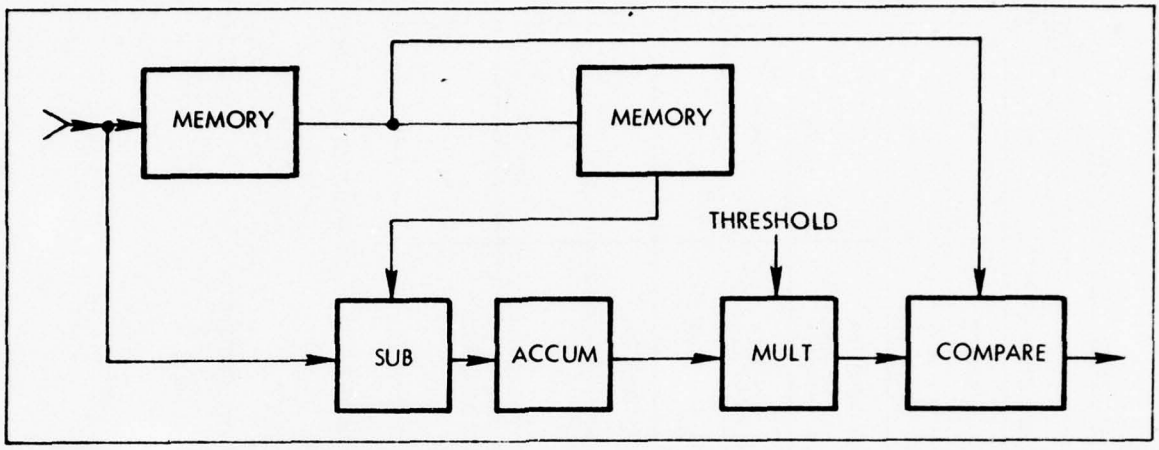


Figure 55 - CFAR

Even more complicated forms of CFAR algorithms exist, but in our experience, they don't drive the design of the arithmetic configuration because their thruput is low. However, for these operations the μSP really excels in system development time, algorithm flexibility, and small size compared with a GP computer or a hardwired digital approach.

3.4.9 Signal Processor Implications

Supplementing the above functional analysis with sample system sizes yields the relative amount of arithmetics and storage required for each algorithm. These are shown in Table 14 Note that despite the varying number of real adds and real multiplies required per input point, the ratio of the two is mostly between one and two adds per multiply. Because of multiplier costs and thruput needs, we recommend that the μSP Arithmetic Unit contain one real multiplier and two real adders. Furthermore, the Arithmetic Unit should use a multiclock macro instruc-

TABLE 14

SAMPLE COMPUTATION REQUIREMENTS DERIVATION

Radar Function	Algorithm	Granularity	Working Memory Size *1	Real Add/Subtracts *2	Real Multiplies *2
Pulse Compression	Barker Code	13 Pts	26 Complex	14	0
	FFT, Mult, FFT	128 Pts	640 Complex	86	52
Spectral Analysis *3	FFT - Small	64 Pts	64 Complex	18	10
	FFT - Large	1024 Pts	1024 Complex	30	18
Roughing Filter	Mix, Recursive Filter, Desample	7 pole, 7 zero 6→1 Decimation	7 Complex	16 1/3	16 1/3
	Mix, FIR Filter, Desample	40 Pt Kernel 8→1 Decimation	41 Complex	10	5
MTI/AMTI	Store, Mult Accum	5 Pulse	4 Complex	8	8
Mag and Integ	ABS VAL, Large-Small, Accum-Delay	4 Pulse	1 Real	5	2
CFAR	Sliding Window Average, Thresh-old	16 Pts	17 Real	3	1

* Note 1: Word quantity is multiplied by the number of concurrently processed channels.

* Note 2: Operations are quantity per input point.

* Note 3: N point FFT's on real data require same memory and arithmetics as $\frac{N}{2}$ point complex FFT.

tion as done in Raytheon's MINI GPSP. This would allow a minimum number of arithmetics to execute the most basic functions such as in Figures 45 through 54, in one pass of the data through the AU. The less frequently called-upon algorithms can take more steps, if need be, to keep the AU design simple without significantly reducing overall performance.

Adding the functions derived in the above analysis to the functions already implemented in Raytheon's programmable signal processors yields the list of desirable algorithms given in Table 15. Elimination of multiple variations of the same type of computation leaves the abbreviated list presented earlier in Table 14.

TABLE 15

GENERAL SIGNAL PROCESSING FUNCTIONS

Vector negate	Complex FFT
Vector add	Real FFT
Vector subtract	Inverse FFT
Vector multiply	Bit reverse order an array
Vector divide	Convolution
Vector-scalar add	Correlation
Vector-scalar multiply	Weiner-Levinson algorithm
Vector normalize	Burg algorithm
Vector rescale	Bandpass filter
Sum of vector elements	Power spectrum
Dot product of 2 vectors	Complex spectrum
Max, element of vector	FIR
Min. element of vector	FIRN
Vector max, magnitude	FIRPZ
Vector min. magnitude	IIR
Complex vector multiply	Cross spectrum
Complex vector reciprocal	Transfer function
Complex vector magnitude	Coherence
Comples conjugate	Sliding window sum
	CFAR

3.5 Wordlength Analysis

3.5.1 Introduction

Usually wordlength studies are directed to a specific application. For the Micro Signal Processor we are attempting to establish bounds on wordlengths for a machine of general capability. In this broader context, we must consider classes of filters and ranges of performance levels. Our task is somewhat eased by implementation consideration because digital words are conveniently available in 4 bit sections.

We shall start with a view of the importance of having a variety of word formats to do computations within a fixed word size. Historically, as digital arithmetic and memory became cheaper, limited bit processing is no longer justifiable in the performance cost tradeoffs, Furthermore, multimode processing requirements add to the need for including multiple format words within a programmable processor. These formats include:

- Single precision complex
- Double presicion integer
- Single precision floating point
- Double precision floating point
- Block floating point

The variable word formatting allows the maximum dynamic range to be carried where needed. More generally, the unit is no longer tailored to one task, but can be applied to many of the functions which are required for the mission.

Word lengths are established on the basis of filter sidelobe levels, and dynamic range. Based on material supplied below, a word length of 12 bits is required to support applications requiring dynamic range and sidelobe levels in the 50 db class. An extended word length of up to 24 bits should also be

available to accommodate those situations where high processing requirements are encountered.

3.5.2 GENERALIZATIONS

3.5.2.1 Fixed Point Single Precision

This is a format used in smaller machines and is usually the smallest word size format. It is used in processing data in conjunction with a predetermined scaling sequence.

Its main use is in algorithms where pair-wise summation take place and interest in the output lies in the extraction of a target from a noise background. An example of such an algorithm is the FFT. The signal to noise improvement of pair-wise summations allows scaling and word truncation to take place in such a way as to maintain the same word size and still extract the targets of interest.

3.5.2.2 Double Precision

This format uses two of the single precision words to represent one large word. In signal processing, since most words travel as complex pairs, the single precision format usually carries two of the vector components. The double precision format uses both these words to carry one vector component with twice the precision.

The main use of the format is where truncation produces unacceptable errors. One such place is where a large accumulation of samples takes place to generate an average. Such an average might be used to measure background noise for the purposes of establishing a detection threshold. The data cannot be truncated or scaled until all points have been summed, if they are to have equal weight and accuracy in their effect on the final answer.

One other case where truncation is unacceptable is in various stages of SAR map processing. The map information in this case contains energy in a wide spectrum of frequencies. It is therefore noise-like in the way it passes through an FFT. There are cases where a strong reflector, such as a water tower, dominates the receiver dynamic range. Such a reflector acts as a false target. We are not interested in this target, but rather in the fine-grain detail of the map -- which looks like noise. We therefore cannot truncate this noise as we previously did when we were only interested in the dominant target. Word growth has to be allowed until noise growth takes place. In large size FFT's this requires double precision in the later stages of the FFT process. This is a similar problem to multiple target detection.

3.5.2.3 Floating Point

This format carries a complex mantissa and a common exponent. Scaling in this format is data dependent, with shifting down and truncating occuring as data grows. One area where this becomes very useful is in weighted accumulate-and-dump filters such a clutter cancellers or FIR filters where we are interested in target detection. Such accumulations would have to allow extensive word growth if we were interested in the accuracy of the answer such as in the noise average. However, when such growth occurs we are dealing with large targets where gain and cancellation accuracy are not important. Such targets are easily detected. Since we are dealing with one cell, the presence of a large target will mask any small targets presence in the same cell. With just small targets present truncation need not occur and cancellation accuracy is therefore preserved.

Another area where floating point is important is in multiplication and division of video data by video data. One such example occurs in monopulse ratio calculation used for target tracking and ground moving target detection. Here the

floating point is used in a normalization mode which moves the significant information toward the most significant bits of the word. This is done prior to mulitplication or division in order to eliminate the need for increasingly large word sizes to hold answers.

3.5.3 Dynamic Range

3.5.3.1 Input Dynamic Range

We treat input dymanic range for completness. The performance of a digital signal processor is more properly controlled by the required output dynamic range. The input dynamic range is set by the A/D converter sourcing the data. Depending on the problem this ranges from 1 to 14 bits.

3.5.3.2 Outupt Dynamic Range

The broadest concept of output dynamic range is set by the input dynamic range coupled with any signal to noise gain incurred during processing. In consideration that signal to noise improvements of 30 to 40 db are not uncommon, we can expect worst case output dynamic ranges of 8 to 21 bits.

The long word lengths developed above should not be accepted as a definitive argument for setting processor wordlengths. There are fundamental limitations as well as systems requirements limitations that allow the practical use of much shorter wordlengths.

3.5.3.2.1 Signal Detection

In many applications, the signal processor is being used to permit detecting a small signal in the presence of noise. Acceptable detection can be achieved when the signal to noise ratio at the output of the signal processor is 10 to 13 db. If we force the quantizing noise generated during the

processing to be small compared to the background noise accompanying the signal, this level of performance can be obtained using 3 or 4 bits.

Of course there will be situations when the signal is substantially larger than the background noise. For this situation we must either provide more bits to handle the larger signal, or we must allow for scaling within the processor to prevent overflow of larger signals.

3.5.3.2.2 Signal Separation

An alternative to the signal detection situation occurs in the signal separation case. In this instance a small signal is to be observed in the vicinity of some larger signal. In this situation the system performance is frequently limited by the capabilities of the waveforms. An example of this type of limitation is furnished by pulse compression systems. In such systems, the main response is centered in a region of sidelobes. The sidelobe level is set by a combination of the time band width product of the waveform, and the weighting function used for sidelobe reduction. The weighting function selection involves trading off resolution and detectability of obtain lower sidelobes. The sidelobe levels sought in practical systems range from 35 to 50 db. Because of these limitations, in the vicinity of a strong target, a 60 db output dynamic range would permit observation of the strong targets sidelobes and any smaller target that exceeded the sidelobes.

3.5.3.2.3 Intense Clutter Background

A most severe processing situation occurs when pulsed doppler radars are used to examine low flying targets from a look down position. Systems of this type can require 60 to 90 db of discrimination against the clutter background. In this situation, the output dynamic range requirements are moderate.

The input dynamic range requirements are more severe because the target signal must exceed the A/D quantizing noise in the narrow bandwidth occupied by the target. The most severe requirement is imposed by filter stop band attenuation which excludes the strong clutter from the detection process. As an example in a system with 60 db clutter to target ratio, and a 1000: 1 bandwidth reduction we have 30 db of processing gain against A/D quantizing noise. To obtain a 15 db signal to quantizing noise ratio a 7 bit A/D is required. In addition a 75 db of stop band attenuation must be provided.

3.5.3.3 Gain Adjustment

In several of the cases examined above, attention was concentrated on making a local observation of a weak target. The aspect of a strong target was not of major concern. Using this outlook, the presence of strong signal require accommodation by some form of gain adjustment, or by increasing wordlength. This implies providing gain adjustment for the receivers, and the allocation of extra bits to the A/D converter before the processor.

The processor accommodation will require the use of fixed point scaling, or some form of full floating point, or block floating point operation.

3.5.4 Processor Wordlength

The major function of a signal processor is filtering. This filtering involves multiplying input signal samples by weighting coefficients and adding these products and other products to obtain the filtered results.

The wordlength required for the weighting coefficients (which affects response shape) can be treated separately from the wordlength used for computing and storage (which affect dynamic range).

3.5.4.1 Weighting Coefficient Wordlength

A filter's response is set by its poles and zeroes. It is possible to separate the examination of zeroes-only filters from filters having only poles.

3.5.4.1.1 Zeroes Only Filters

Chan and Rabiner [1] resolved the coefficient wordlength requirements of a zeros only (FIR-Finite Impulse Response) filter with the relation that the inband rejection is given by

$$DLk^{*} \tilde{\geq} -20 \text{ LOG } \left[10^{DLk/20} + 2^{-t}\sqrt{\frac{2N-1}{3}} \right] \quad (1)$$

where t is the word length exclusive of sign, N the number of samples in the impulse response, DL_K the desired in band rejection with ideal implementation and DL_{K*} the achieved inband rejection. The symbol $\tilde{\geq}$ means that most of the time this level of performance will be achieved. Since this relation is based on the round off statistics occasionally the results will be worse than predicted.

Inband rejection has the obvious meaning in the filter stop band, i.e. how many db of rejection do you get in the stop band. In the pass band, it is a measure of the deviation of the response from the desired response. Based on this concept, we can construct Table 16.

In consideration that stop band attenuation of 40 to 60 db is frequently required we can observe that the stop band performance will generally set the required number of bits for the coefficients.

TABLE 16 - CONVERSION OF PASS BAND RIPPLE TO IN BAND REJECTION

In Band Ripple	In Band Deviation	In Band Rejection
db	db	db
0.2	0.1	-38.8
0.5	0.25	-30.7
1.0	0.5	-24.5
3.0	1.5	-14.5

Figure 56 was constructed from eqn.(1) using N equal to 32 to show the effect of different coefficient wordlengths on stop band rejection. N=32 was used because experience has indicated that filters having very long impulse responses can be implemented using multiple filter stages with sample thinning between stages [2]. From Figure 57 we can observe that 40 db performance can be achieved with 50 db design and 10 bit weights, 50 db performance with a 60 db design and 12 bit weights, and 60 db performance with a 70 db design and 14 bit weights.

3.5.4.1.2 Filters Having Poles

For filters having poles, we refer to two reasonably high performance filter designs as examples on which to base our judgements.

The first example was discussed by Crochiere (16). This filter is an 8 pole, bandpass, elliptic filter with the following characteristics.

Bandwidth/fo	5%
Transition width/fo	25%
Pass band Ripple	0.5db
Stop band Attenuation	40db

with fo the sampling rate

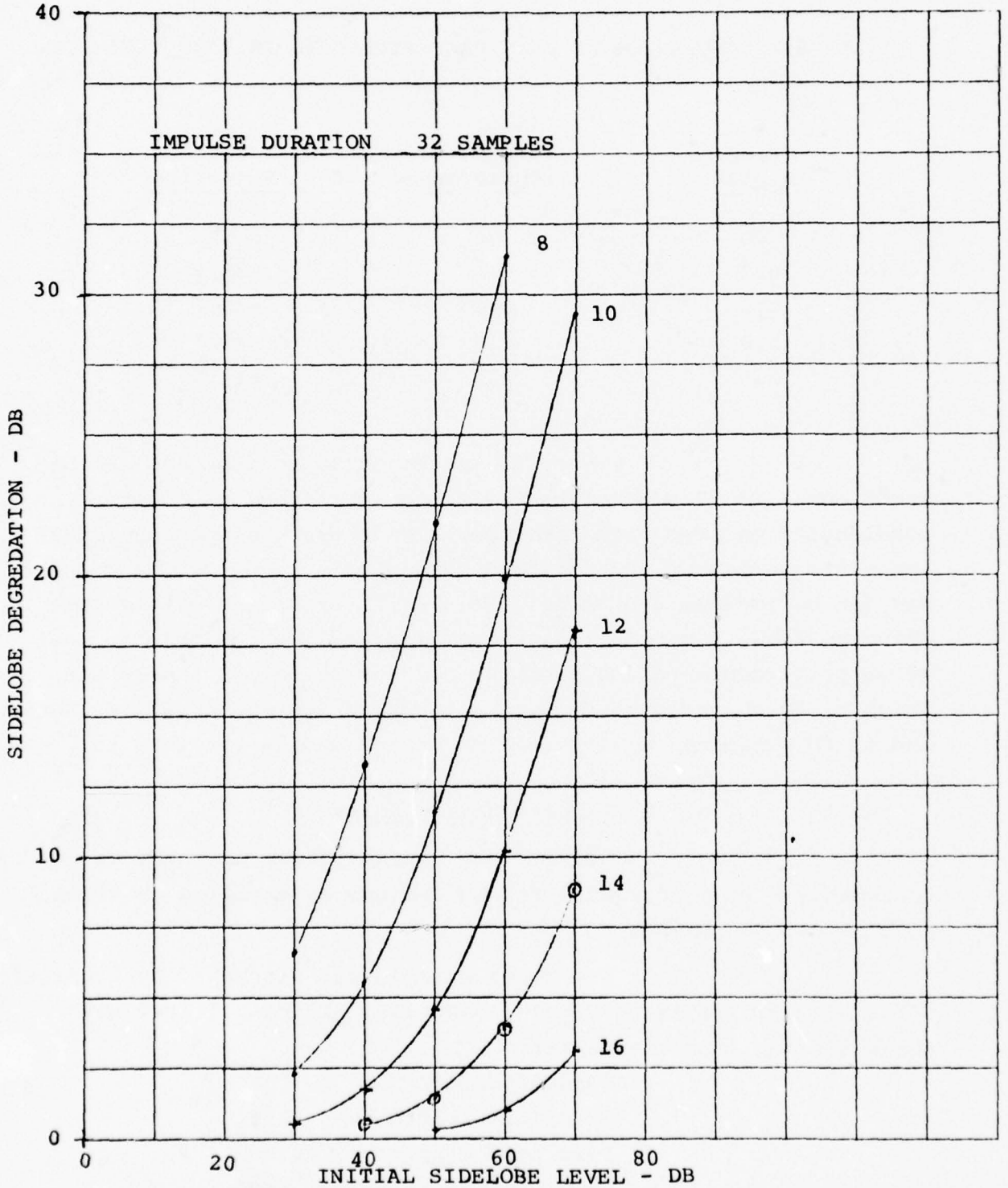


Figure 56 - Coefficient Wordlength vs Stop Band Rejection

Several different structures were used to implement this filter, and it was concluded that

(a) Maintaining passband flatness required more coefficient accuracy than maintaining stop band attenuation.

(b) The structure using a cascade of 2-pole sections required 9 bit coefficients.

An example from Raytheon experience was an 8 pole low pass elliptic filter with the following characteristics:

Band width/fo	5.5%
Transition width/fo	3%
Pass band Ripple	0.5db
Stop Band Attenuation	75db

This filter was implemented as a cascade of 2-pole sections and required 10 bit coefficients to achieve the desired performance levels.

Although these two filters do not represent a full range of filter requirements they do indicate that 10 bit word lengths can provide filters with poles that produce results that achieve high quality performance.

3.5.4.1.3 Coefficient Wordlength Recommendation

For filtering systems, based on 4.1.1 and 4.1.2, and in consideration that wordlengths come in convenient 4-bit sections, a 12 bit coefficient word length would provide an adequate range of capability.

3.5.4.2 Wordlength for Computation

During processing, wordlengths will be reduced after multiplication, These round off operation inject noise

into the system that propogates to the output. As with the case for coefficient wordlengths, we separate the zeros-only filter from the filter having poles.

3.5.4.2.1 Round Off Noise in Zeros Only Filters

There are two situations of interest here. The first case is that of direct form implementation. In the direct form, each round off produces noise which propogates to the output with unit gain. For an N stage linear phase filter, there are N/2 round off multiplications. These produce an output noise variance:

$$\delta_o^2 = \frac{N}{2} \delta_\epsilon^2 \quad . \qquad (2)$$

For a b-bit word including sign:

$$\delta_\epsilon^2 = \frac{2}{12}^{2(b-1)} \quad . \qquad (3)$$

The peak signal the filter output can have is unity. Combining these, the dynamic range of the filter output becomes

$$DR = 10 \text{ LOG } \left[\frac{2^{2(b-1)}}{N} \times 24 \right] \qquad (4)$$

Evaluation of (3) is shown in Table 17

TABLE 17

DYNAMIC RANGE OF ZEROS ONLY FILTERS

N	10	20	40
b			
8	46	43	40
10	58	55	52
12	70	67	64
14	82	79	76

For the second case we treat a cascade filter arrangement where noise generated in the earlier stages is filtered by the action of subsequent stages. We choose for this example the FFT, because of its wide usage. We examine the highest frequency term of the FFT because it has the most multiplies. Figure 57 shows its flow chart in pruned form and includes injected noise powers of $q^2/3$ at each of the nodes, and noise power gains of $1/2\ q^2/3$ was used because four multiplies are used in the complex twiddle.

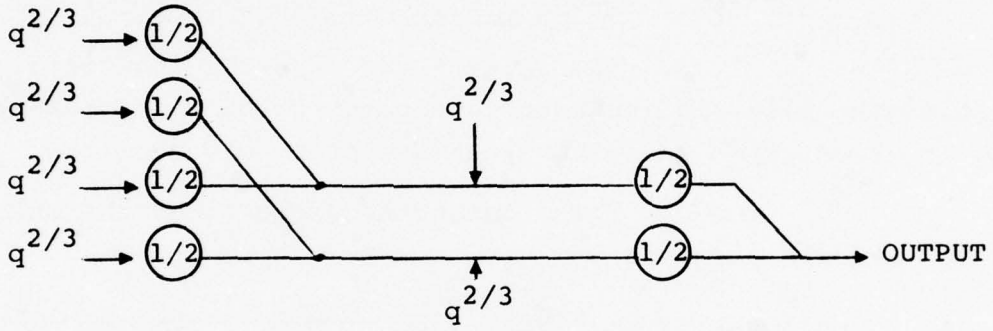


Figure 57 - FFT Pertinent to Noise Analysis

From Figure 57, the output noise power is

$$N_{out} = \left(\frac{1}{2} \times \frac{2q^2}{3}\right) + \left(\frac{1}{4} \times \frac{4q^2}{3}\right) \qquad (5)$$

Extending this result, and taking the maximum output of the FFT as unit we have

$$DR = 10 \log \frac{3 \times 2^{2(b-1)}}{\log_2 N} \qquad (6)$$

with N the number of points in the FFT and b the word length. Evaluation of (5) is shown in Table 18.

TABLE 18

DYNAMIC RANGE OF FFT WITH 1/2 POWER SCALING

N	8	16	32	64
b				
8	42	41	40	39
10	54	53	52	51
12	66	65	64	63
14	78	77	76	75

3.5.4.2.2 Round off in Filters with Poles

The filters use in examining coefficient wordlengths will also be used in examining round off. We take the implementation as being a cascade of 2-pole sections.

For a second order section, the output noise variance is given by [7],

$$\delta_o^2 = \frac{2}{12} \; 2^{-2(b-1)} \; \frac{1+x^2}{1-x^2} \; x \; \frac{1}{x^4 + 1 - 2x^2 \cos 2\theta} \qquad (7)$$

Where $x\theta$ is the polar coordinates of the poles. The pole radius is limited to unit value for a stable filter.

Equation (6) indicates that the poles with the largest value of r produce the greatest output variance. In a cascade arrangement, the poles with small radius should be placed at the end of the filter to reduce the effect of large pole sections by filtering. In this arrangement, the output noise will be dominated by the contribution of the last section.

With this choice, the results of Table 19 were computed for the Crochere and Raytheon filters.

TABLE 19

DYNAMIC RANGE OF TWO FILTERS BASED ON LAST STAGE

	Crochere Filter	Raytheon Filter
b		
8	42.1	30.5
10	54.1	42.6
12	66.2	54.6
14	78.2	66.6

3.5.4.2.3 Process or Wordlength Recommendation

For filtering systems, based on 3.5.4.2.1 and 3.5.4.2.2, and in consideration that wordlengths come in convenient 4 bit sections, a 12 bit processing wordlength is recommended.

3.5.4.3 Severe Requirements

It is to be anticipated that some applications will require greater performance levels than those assumed here. In anticipation of these requirements, multiple precision operations should be available in the system.

3.5.5 References

(14) D.S.K. Chan and L.R. Rabiner
IEEE Tran. Audio Electracoust., Vol. AU-21, pp 354-366, August 1973.

(15) R.E. Corchiere and L.R. Rabiner
IEEE Tran. Acoust., Speech, Signal Processing
Vol ASSP-23, pp 444-456, Oct. 1975

(16) R.E. Corchiere and A.V. Oppenheim
Analysis of Linear Digital Networks
Proc. IEEE, Vol 63, pp. 581-595, April 1975

(17) A.V. Oppenheim and R.W. Schafer
Digital Signal Processor

Prentice Hall 1975

SECTION IV
STATE-OF-THE-ART SURVEY

4.1 Introduction

This task involved a survey of logic building blocks and their trends to postulate compatible micro signal processor design features. The objective is to maximize the use of commercially available and compatible LSI chips for now and some few years to come. Many surveys have appeared recently trying to project IC developments, such as the article: Trends in Computer Hardware Technology, by D. Hodges in the February 1976 issue of Computer Design. The key elements we have superimposed on these technology predictions are our past experience and our present healthy skepticism.

Standard building blocks began emerging in the MSI era of ICs. Exactly the same function could be obtained independently of the technology family desired, whether it be CMOS, TTL, LS/TTL to ECL 10K. Examples were selectors, hex D flops, counters, arithmetic logic units (ALU), scratchpads and priority encoders. Standardization allowed the manufacturer lower development costs and risk, higher volume while giving the user lower prices, familiar functions, shorter design cycle and intelligible design.

Today a number of standard functions are emerging in the LSI technology, although their design details are not 100 percent identical. Examples are the CPU slice with its register file and ALU, sequencers, input-output bus/communicators, first-in first-out queue's (FIFOs), last-in first-out queue's (LIFOs or Stacks), and high density RAMs, ROMs, PROMs, PLAs and FPLAs. These offerings will be judged on their survival likelihood as well as their usefulness.

The focus of this survey is on analysis of features and trends, rather than on mountains of raw comparisons. The latter is typified by the selection of micro processor references given in Table 20.

TABLE 20
SELECTED MICRO PROCESSOR REFERENCES

(18) Eugene Hnatek, "Chipping Away At Core," Digital Design, July 1976, p. 31-42.

(19) Jean Nicoud, "Peripheral Interface Standards For Microprocessors, Proc, IEEE, June 1976, p. 896-904.

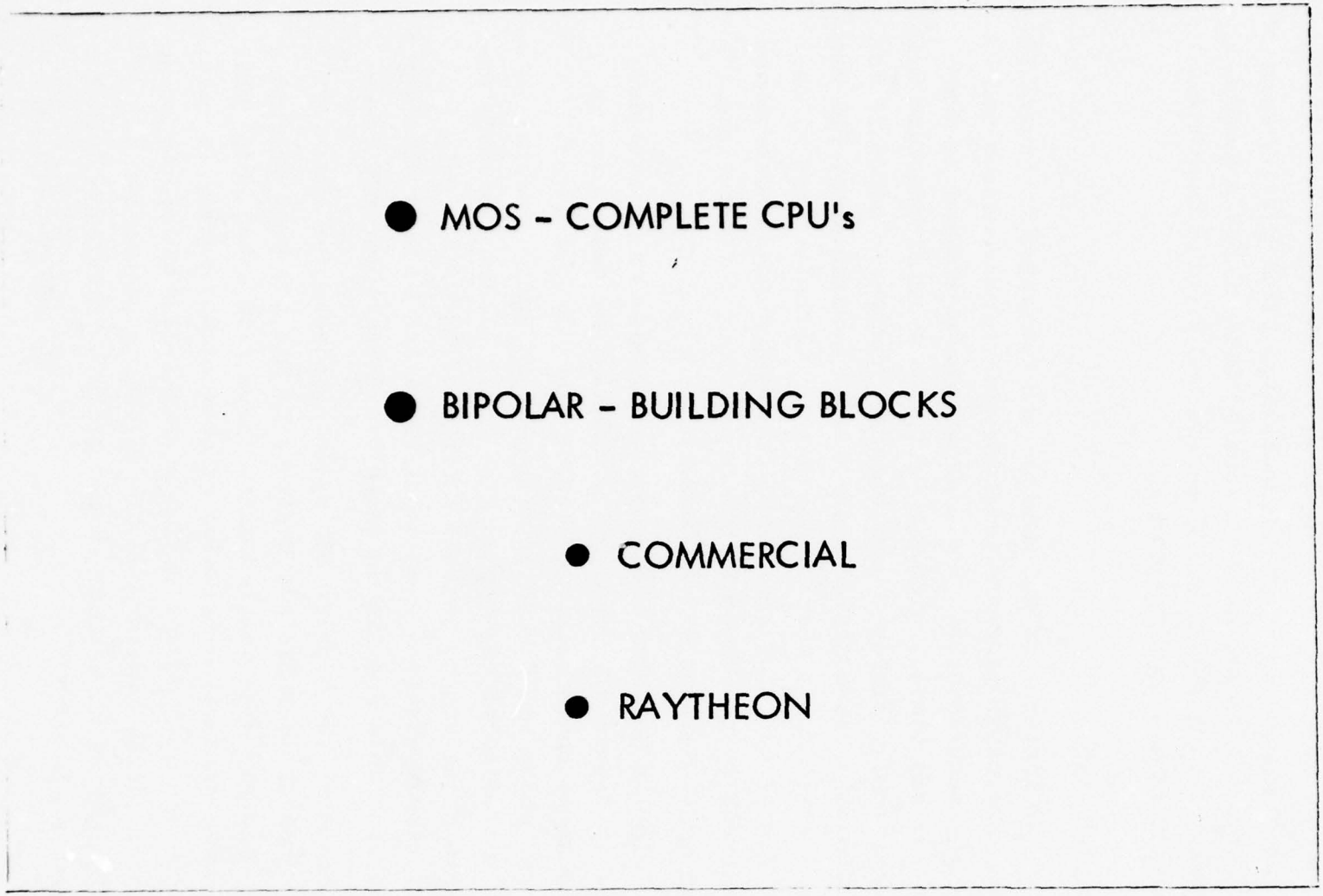
(20) A. Williams & H. Jelinek, "Introduction to LSI Microprocessor Developments," Computer, June 1976, p. 34-46.

(21) EDN MICROCOMPUTER SYSTEMS DIRECTORY, Cahners Publishing Co., 1975,

(22) MICROPROCESSOR SCORECARD, Microcomputer Techniques, Reston Virginia, Mini Micro Systems, July 1976.

(23) MICROCOMPUTER D.A.T.A. BOOK EDITION 1, 1976, D.A.T.A. BOOK INFORMATION SERVICE, Orange, N.J.

The design philosophy here is to employ the best mixture of available semiconductor technology (Figure 58). The MOS devices form complete CPU's and will be postulated for slower speed peripheral control and communication tasks. The commercial bipolar LSI building blocks are to be employed for the high speed heart of the micro signal processor. Functional areas where significant performance, size or availability improvements exist



Fiqure 58 - Available Semiconductor Technology

over commercial LSI will be considered for implementation by Raytheon's programmable gate array capability. This partitioning of application areas is expected to remain constant for a number of years, even considering that MOS speeds and bipolar densities will continue their upward spirals.

4.2 General Trends

Some key features of MOS devices are presented in figure 59. The fixed instruction formats preclude most architectures not given by the manufacturer. MOS microprocessor thruput is low (microsecond add times), although performance on instruction mixes may be drastically improved when special arithmetic hardware is added, such as a bus-oriented multiplier. Nevertheless, MOS microprocessors exist for a host of peripheral controller tasks, as shown in figure 60. For most of those tasks, speed improvements are irrelevant to the mechanical or human factor limitations. A notable exception is interval timing.

The bipolar building blocks offer a number of advantages as listed in figure 61. Most significantly, we can configure them into an advanced architecture for high thruput signal processing, while still meshing with MOS LSI for peripheral interfacing. A preview of the types of building blocks available today or within the coming year is shown in figure 62. Note that some fast bipolar devices such as 8K PROMs and 64 word FIFO's have been available for two or three years in slower MOS forms.

In choosing the bipolar LSI building blocks for the Micro Signal Processor we shall not hesitate to mix the best designs from one chip set with complementary chips from other chip sets. For immediate production-oriented efforts such a policy is hazardous, since one must depend on the survival of several competing design sets. Here, however, we are selecting particular LSI IC's only as indicative of a trend toward certain function blocks and

- FIXED INSTRUCTION FORMATS
- SUPPORT SOFTWARE SUPPLIED BY VENDOR/SOFTWARE HOUSES
- SUITABLE FOR LOW-THRUPUT REQUIREMENTS (3 μsec ADD)
- THRUPUT UPGRADEABLE BY PERIPHERAL ARITHMETIC HARDWARE
- FULL COMPLEMENT OF PERIPHERAL CONTROLLERS AVAILABLE

Figure 59 - MOS CPU/Microprocessor's

- FLOPPY DISC CONTROLLER
- MAGNETIC TAPE CASSETTE CONTROLLER
- CRT CONTROLLER
- KEYBOARD AND DISPLAY CONTROLLER
- DYNAMIC MEMORY CONTROLLER
- INTERVAL TIMER
- UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER
- MODEM INTERFACE

Figure 60 - MOS Controllers

- ALLOWS USER TO DESIGN PROPRIETARY PROCESSOR
- REQUIRES USER TO DEVELOP OWN SUPPORT SOFTWARE
- PROVIDES HIGH THRUPUT (0.3 μsec ADD)
- CONFIGURABLE INTO HIGHER THRUPUT SIGNAL PROCESSORS
- MAY INTERFACE WITH MOS LSI PERIPHERALS

Figure 61 - Bipolar LSI Building Blocks

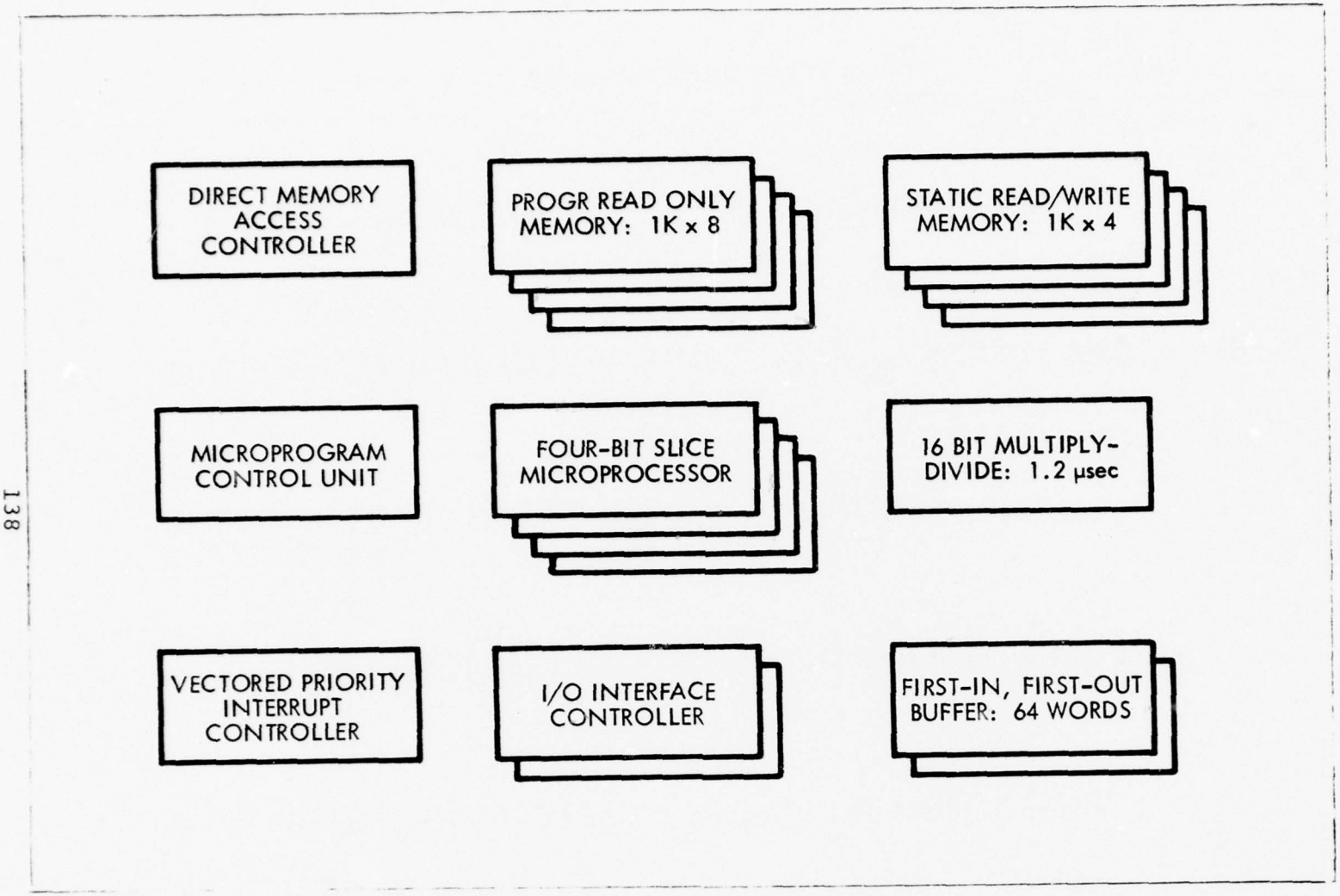


Figure 62 - Selected Bipolar LSI Building Blocks

including certain features. We expect that the surviving bipolar LSI family (or families) will eventually incorporate the most marketable aspects of their fallen competitor's designs. An overview of the present LSI Microprocessor Families is given in Table 21. The twenty entries listed in the table exclude inappropriate designs such as the slower PMOS types, the older 4 bit microprocessors, and a few false starts such as Intel's 8008 and 3000 series. A variety of technologies still remain, depending on tradeoff of speed, low power, chip density and manufacturing costs. Clear trends include:

- Domination and interrelation of 8 and 16 bit machines; where most 8 bit machines have 16 bit address fields, and some 16 bit machines actually process data in 8 bit pieces.

- Bipolar slices come in 4 bit widths

- MOS device speeds are close to each other, but an order of magnitude slower than bipolar speeds.

- Inclusion of significant amount of registers (RAM) within the CPU, with 16 words common and some 64 words.

- Operation with only one 5 volt power supply

- Recognition of the need for multiple sources, with the 8080 and 2900 leading with 5 and 4 sources respectively.

- A large number of microprocessor's are really versions of other computers, including Intersil's 6100 (PDP-8), TMS9900 (TI990), Data General micro NOVA, and Western Digital's 1600 (PDP-11).

TABLE 21

MICROPROCESSOR FAMILY SRUVEY

MICROPROCESSOR FAMILY	TECHNOLOGY	DATA BITS-CONTROL-ADDRESS	REG ADD TIME (μsec)	QTY CPU REGISTERS	VOLTAGES	LOW POWER	SECOND SOURCES
RCA COSMAC CDP1802	CMOS	8-8-16	2.5	16	3 TO 15	**	
INTERSIL 6100 (PDP-8)	CMOS	12-12-12	2.5	1	5	**	HARRIS
SIGNETICS 2650	NMOS	8-8-15	5.0	15	5		AMS, MULLARD
FAIRCHILD F8	NMOS	8-8-16	2.0	65	5, 12		MOSTEK
MOTOROLA 6800	NMOS	8-8-16	2.0	3	5		AMI, HITACHI
MOS TECHNOLOGY 6502A	NMOS	8-8-16	1.0	3	5		SYNERTEK
INTEL 8080A	NMOS	8-8-16	1.3	7	-5, 5, 12		AMD, TI, NEC
ELECTRONIC ARRAYS 9002	NMOS	8-8-12	2.0	79	5		
ZILOG Z-80	NMOS	8-8-16	1.6	16	5		
GENERAL INST CP-1600	NMOS	16-16-16	2.4	6	-3, 5, 12		EMM
DATA GENERAL μN601	NMOS	16-16-15	2.4	5			
TEXAS INSTRUMENTS TMS9900	NMOS	16-16-15	2.7	1	-5, 5, 12		
WESTERN DIGITAL 1600 (PDP-11)	NMOS	16-16-16	0.6	16	-5, 5, 12		
TEXAS INSTRUMENTS SBP0400	IIL	4N	0.5	9	1	*	
SCIENTIFIC MICRO SYS SMS300	TTL	8-16-13	0.3	8	5		SIGNETICS
MONOLITHIC MEMORIES 6700'	TTL	4N	0.2	17	5		
ADVANCED MICRO DEV 2900'	TTL	4N	0.13	17	5		RAY, MOTOROLA
TEXAS INSTRUMENTS 54S481	TTL	4N	0.10	1	5		
FAIRCHILD MACROLOGIC	TTL	4N	0.08	8	5		SIGNETICS
MOTOROLA 10800'	ECL	4N	0.06	1	-5, 2, -2		

- Improvements on existing microprocessors are occuring rapidly including the Z80 (on the 8080) and the 6502 (on the 6800)

4.3 LSI IC Features

A detailed examination of specific LSI chip types was made and compared in Table 22, Table 23 and Table 24. Table 22 lists bipolar sequencer types, followed by CPU slice types. Table 23 covers MOS CPU types. Table 24 covers MOS peripheral controller types. Significant effort was put into understanding the features available across a host of MOS devices in order to postulate features likely to be seen in a few years in higher speed building blocks. The presence within an LSI IC of significant logic segments for performing an identified computer task forms the features listed on the left side of the tables. These features are clarified in the remaining paragraphs of this section.

Bit Width represents the number of data bits which an IC appears to handle in parallel to the outside world. Bipolar sequencers are either 4 bit slices or a fixed number of bits (up to 10 today). RALU slices are almost overwhelmingly 4 bits wide. Serious interest in 8 or 16 bits is exhibited in the MOS world.

Program Address Counter is present in all sequencers and MOS CPU's but rarely in peripheral controllers or RALU slices. Most exceptions are caused by the F8 family, which distributes the address function to some bus communication.

Program Stack is really a Last-In, First-Out (LIFO) queue. Its primary function is to store subroutine return addresses. Stack sizes vary from 4 to 16 words, with clear dominance of 4 words in bipolar sequencers. Most MOS CPU's achieve this function in other ways such as software pointers to main memory stack. We thus expect to see a continuation of stacks in bipolar sequen-

TABLE 22

MICRO SIGNAL PROCESSOR LSI CHIP SURVEY SHEET I

LSI IC FEATURES \ LSI PART NUMBER	2909/11 AMD	5482 TI	6710 MMI	3001 SIG	8X02 SIG	9408 FAIR	9406 FAIR		3002 SIG	2901 AMD	9405 FAIR	6701 MMI	6702 MMI	10800 MOT	SBP0400 TI	5481 TI	8X300 SIG			
BIT WIDTH	4	4	8	9	10	10	4		2	4	4	4	4	4	4	4	8			
PROGRAM ADDRESS COUNTER	X	X	X	X	X	X	X								X		X			
PROGRAM STACK	X 4x4	X 4 WORD			X 4x10	X 4x10	X 16x4													
MULTI-WAY PC INCREMENT		X	X	LOGIC JUMPS	X	X	X													
INSTRUCTION DECODE			X	X	X	X			X	X	X	X	X		X		X			
PROGRAM ROM																				
STATUS/FLAGS/INTERRUPT FF's & LOGIC			X	X		X														
ARITHMETIC LOGIC UNIT & SHIFTER									X	X	X	X	X	X	X	X	X			
REGISTER FILE (BEYOND MISC. REGISTERS)/RAM									X 9x2	X 16x4 2 PORT	X 8x4	X 16x4	X 16x4		X		X			
SHIFT REGISTER (APART FROM ALU REG.)										X		X		X		X	X			
ADDRESS OUT PORT	X	X	X	X	X	X	X		X						X		X			
I/O BI-DI BUS PORTS													X 1x4	X 2x4						
HOLD CLOCK LINE																				
RESET INPUT	X			AUTO RESET ON POWER UP	X												X			
CLOCK (MINUS ONLY XTAL)																				
MULTI INTERRUPT LOGIC/REG																				
ADDRESS COMPARER																				
INTERVAL TIMER																				
SHIFT CYCLE COUNTER			X																	
FANCY SR I/O; PARITY, SYNC...																				
DATA MASK REGISTER/INPUT									X					X			X			
KEYBOARD/DISPLAY MECHANISM:																				
IC PINS	28 DIP	20 DIP	40 DIP	40 DIP	28 DIP	40 DIP	24 DIP		28 DIP	40 DIP	24 DIP	40 DIP	40 DIP	48 QUIL	40 DIP		50 DIP			
CATEGORY	SEQUENCERS						LIFO		RALU's (CPU SLICES)											

TABLE 23

MICRO SIGNAL PROCESSOR LSI CHIP SURVEY SHEET II

LSI PART NUMBER / LSI IC FEATURES	4040 MCS40 INTEL	PPS4/1 ROCK	11660 PPS4/2 ROCK	TMS1000/1400 TI	TMS1100/1300 TI	EA9002 ELEC ARR	3850 F8 FAIR	8080A MCS80 INTEL	MCS6502 MOS TECH	MCS6503/04/05 MOS TECH	M6800 MOT	SC/MP NAT SEMI	CDP1801 RCA	CDP1802 RCA	10806 PPS8 ROCK	2650 S/G	IM6100 INTERSIL	CP1600A GEN INST	IPC16A500 NAT SEMI	TMS9900 TI
BIT WIDTH	4	4	4	4	4	8	8	8	8	8	8	8	8	8	8	8	12	16	16	16
PROGRAM ADDRESS COUNTER	X	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X	X
PROGRAM STACK						X 7x12										X 7x15			X 10 WORD	
MULTI-WAY PC INCREMENT		X	X					X			X		X	X		X				
INSTRUCTION DECODE	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
PROGRAM ROM	X	X 1.3Kx8	X 2Kx8	X 1Kx8	X 2Kx8														X MICRO	
STATUS/FLAGS/INTERRUPT FF's & LOGIC	X	X	X			X	X	X	X	X	X	X	X	X		X		X	X	X
ARITHMETIC LOGIC UNIT & SHIFTER	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
REGISTER FILE (BEYOND MISC. REGISTERS)/RAM	X 8x12 24x4	X 96x4	X 128x4	X 64x4	X 128x4	X 8x8 64x8	X 64x8	X 6x8			X 2x8	X	X 16x16	X 16x16	X	X 64x8		X	X	
SHIFT REGISTER (APART FROM ALU REG.)		X	X				X					X						X		X
ADDRESS OUT PORT						X		X	X	X	X	X	X	X	X	X				X
I/O BI-DI BUS PORTS		X 1x4	X 1x4			X 1x8	X 3x8	X 1x8	X 1x8	X 1x8	X 1x8	X 1x8	X 1x8	X 1x8		X 1x8	X 1x12	X 1x16	X 1x16	X 1x16
HOLD CLOCK LINE								X												X
RESET INPUT	X	X POWER ON		X	X	X	X	X	X	X	X	X	X	X	X POWER ON	X	X		X	X
CLOCK (MINUS ONLY XTAL)		X	X	X	X		X					X					X			
MULTI INTERRUPT LOGIC/REG														X			X		X	X
ADDRESS COMPARER																				
INTERVAL TIMER																				
SHIFT CYCLE COUNTER																				X
FANCY SR I/O; PARITY, SYNC...																				
DATA MASK REGISTER/INPUT																				
KEYBOARD/DISPLAY MECHANISM:																				
IC PINS	24 DIP	42 FLAT	42 FLAT	28/40 DIP	28/40 DIP	28 DIP	40 DIP	40 DIP	40 DIP	28 DIP	40 DIP	40 DIP	28/40 DIP	40 DIP	42 FLAT	40 DIP	40 DIP	40 DIP	40 DIP	64 DIP
CATEGORY	MOS CPU's																			

TABLE 24

MICRO SIGNAL PROCESSOR LSI CHIP SURVEY SHEET III

LSI IC FEATURES \ LSI PART NUMBER	3854 F8 FAIR	3851 F8 FAIR	3852 F8 FAIR	3820 F8 FAIR	4707/09/11 MCS40 INTEL	4265 MCS40 INTEL	8255 MCS80 INTEL	8251 MCS80 INTEL	6530 MCS6500 MOS TECH	6850 M6800 MOT	6852 M6800 MOT	MN1630 PFL16A PANAFCOM	10788 PPS4 ROCK	10814 PPS4 ROCK	10817 PPS4 ROCK	SMS360 SCI MIC				
BIT WIDTH																				
PROGRAM ADDRESS COUNTER	X	X	X											X						
PROGRAM STACK		X	X																	
MULTI-WAY PC INCREMENT	X	X	X																	
INSTRUCTION DECODE	X				X	X			X	X			X	X	X	X				
PROGRAM ROM		X 1Kx8	X						X 1Kx8											
STATUS/FLAGS/INTERRUPT FF's & LOGIC		X		X						X	X	X								
ARITHMETIC LOGIC UNIT & SHIFTER																				
REGISTER FILE (BEYOND MISC. REGISTERS)/RAM									X 64x8				X		X					
SHIFT REGISTER (APART FROM ALU REG.)										X 2	X 2									
ADDRESS OUT PORT	X		X												X					
I/O BI-DI BUS PORTS	X 1x8	X 2x8	X 1x8	X 3x8	X 4x4	X 4x4	X 3x8 2x4	X 1x8	X 3x8	X 1x8	X 1x8	X 2x8			X 1x8	X 1x8				
HOLD CLOCK LINE																				
RESET INPUT				X				X	X		X									
CLOCK (MINUS ONLY XTAL)																				
MULTI INTERRUPT LOGIC/REG																				
ADDRESS COMPARER											X					X				
INTERVAL TIMER		X							X											
SHIFT CYCLE COUNTER																				
FANCY SR I/O; PARITY, SYNC...										X	X	X								
DATA MASK REGISTER/INPUT																				
KEYBOARD/DISPLAY MECHANISM:													X	X						
IC PINS	40 DIP	40 DIP	40 DIP	40 DIP	28 DIP	28 DIP	40 DIP	28 DIP	40 DIP	24 DIP	24 DIP	40 DIP	42 FLAT	42 FLAT	42 FLAT	24 DIP				
CATEGORY	PERIPHERAL CONTROLLERS																			

cers as long as a sequencer function is done on a distinct chip.

Multi-Way Program Counter Increment indicates that address sequencing can occur in more than just "current + 1" or "jump to address input." This feature is important for efficient multi-way branching, and can eliminate much of the need for look ahead on nested loops in signal processors. Note that the Intel 3001 is the only bipolar sequencer which doesn't provide address incrementing, but instead uses jumping by logical bit masking. Hence, we conclude that the jumping sequencer, whatever its technical merits, is not going to be the mainstream approach. Multi-way program counter incrementing is less important for MOS microprocessors, where the instruction set is already fixed.

Instruction Decoding varies in complexity from a few gates to a full-fledged PROM or PLA decoding logic. The trade-off is between execution time and generality versus minimum number of control pins and control PROM bits. The control decoder in most MOS microprocessors is primarily devoted to the ALU control, thereby specializing the arithmetic operations to the associated fixed instruction set. For this project we would prefer a more general device having extra speed, and pay the price in extra bits and pins.

Program ROM is now found only in selected MOS microporcessor chips which are oriented toward making systems with an absolute minimum number of chips. Sizes are 1-2K words by 8 bits. Because of volume production considerations, we expect microprocessor's with PROM or EROM included to be developed, but not to replace separate PROM/ROM IC's. Having such a program memory converts a general microprocessor into a special peripheral controller element.

Status/Flags/Interrupt Flip-Flops and Logic is the minimum amount of miscellaneous interchip communication needed to make a working system. All MOS IC's incorporate some of this, but only a few bipolar sequencers have these logic odds and ends. Use of unpersonalized arrays allows signal processor designs to be independent of any one manufacturer's approach to the design and partitioning of this function need.

Arithmetic Logic Unit and Shifter is the key element to differentiate a sequencer chip from a CPU slice chip, or a microprocessor from a peripheral control chip. Note that the shifters are always only one bit up or down, and never the full shift-barrel desired for either true floating point or at least block scaling.

Register Files are clearly going to grow from the 16 words found in some RALU's to the 64 and 128 words already seen in some MOS units. Signal processors can benefit from this not in the data area where megabits of RAM are likely, but in the addressing and subroutine parameter areas. Benefits to those areas include eliminating several external IC's for tasks needing only a small amount of working storage, and faster interrupt handling mechanism by activating separate register segments.

A Shift Register apart from the RALU allows faster instruction execution, such as multiplication and shifting. This feature is not dominant, but useful enough and prevalent enough to postulate as included in our ideal building block.

An Address Output Port separate from the I/O Bus ports is a desirable, but expensive feature. Four and sixteen bit MOS CPU's sacrifice it to pin limitations, while most 8 bit microprocessors have it. Bipolar sequencers all have it, but only some RALU's have it. To be of significant benefit, this address

output port must be tri-state, and operable simultaneously with altering the data port setting. I/O bus ports are a feature which comes in all combinations of sizes in MOS peripheral IC's, and maybe should be done in that manner so long as high speed is not needed.

Clock, Reset and Hold Clock are features that save external miscellaneous logic. We expect that the current multi-phase clock input to MOS chips will pass away, but that clocks will not be included within the high speed bipolar chips for quite a while. An automatic reset when power is turned on, such as found in Signetics 8X02, is very nice for sequencers to start in a predetermined location, but irrelevant for a signal processor's data paths. Holding the clock line allows use with varying memory access timing or instruction execution cycles, but today is available only for the 8080, hardly a trend.

The remaining features, such as interval timing, fancy shift register I/O, are found primarily on special IC types.

4.4 Arithmetics

The available high speed CPU slices, such as the 2901, appear to be very useful for the micro signal processor control elements, but not suitable for the micro signal processor's arithmetic unit. These RALU chips need more input ports, a pipelining orientation, and shifting immediately following arithmetics before output at the very least. The large quantity of storage words within the chip is largely unused for operations involving only two or three complex vectors.

Progress is being made in the desired direction. Variations on the 4-bit slice RALU are now appearing, including MMI's pipeline type, TI's '481 with more ports, and Motorolla's 10800. Such variations on an accepted design may be more acceptable to the marketplace than drastically different CPU slice concepts. Furthermore, where a few mask routing changes can

convert an existing design to a more desirable form, the cost of improvement is small compared with starting from scratch. The most useful such variation is AMD's multi-port RAM, which is just a portion of their popular 2901.

The arithmetic format preferred within the micro signal processor is two's complement. Sign magnitude allows greater ease in mechanizing normalization, scaling, magnituding, and multiplying while two's complement allows greater ease in addition/ subtraction, double-precision expansion and commercial LSI compatibility. Having built programmable signal processors with both formats in the past, we recommend two's complement for LSI compatibility.

A key arithmetic task is multiplication, which often takes as much as 1/6 of the total signal processor ICs. By comparative surveys of alternative multiplication schemes, such as shown in Table 25, the following observations can be made:

- A micro processor is inherently inefficient as a multiplier
- Serial-parallel multipliers are extremely suitable to LSI because of the small number of outputs, and timing domination by internal loop sizes
- Eventual availability of a bus-oriented multiplier will add considerably to the capability of even slow micro processors, but will not create a micro signal processor
- A true combinatorial-logic multiplier would be nice, but the most popular sizes would be unsuitable for signal processing. Our experience has been that multiplier and multiplicand do not need to be the same size, but a 12 bit by 12 bit still would be suitable for all fast applications. Slower processing can then give double-precision capability. Yet the

commercial world tends to think either of 8 by 8 or 16 by 16.

TABLE 25

SURVEY OF DIGITAL MULTIPLIERS

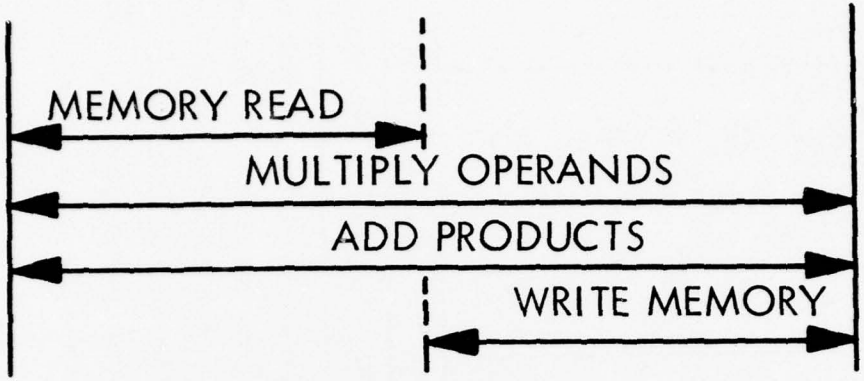
Approach	Size	IC Qty.	No. Clocks	Time Per Multiply nsec	Rel. Efficiency	Comments
Combinatorial Logic = Ideal	8 x 8	1-34 pins ≅ 3	1	50	1	Not yet Commercial
Bus-Oriented of MMI	16 x 16	1-24 pins ≅ 2	12	100	1/4	9-12 Mo. Away
Ser. Parallel AMD 25LS14	8 x 8	5-16 pins ≅ 5	8	50	1/13	Better Design Possible
Combinatorial AMD 25LS05	8 x 8	8-24 pins 2-16 pins ≅ 20	1	150	1/20	Simplest, Multi Source
Micro Proces. AMD 2901	8 x 8	2-40 pins 1-16 pins ≅ 11	12	100	1/60	Useful if Few Mult's.

Note: $$\text{Rel. Effic.} = \frac{3}{\text{IC Qty}} \times \frac{8 \times 8}{\text{Size}} \times \frac{1}{\text{No. Clocks}} \times \frac{50}{\text{Time Per Multiply}}$$

The set of TRW multipliers recently announced are an interesting item to watch. The key question is whether the commercial semiconductor vendors will second source that device approach. Problem areas include a different process than the industry mainstream, several times more power per chip than the usual maximums, and speeds on the slow side of bipolar clock cycles. We do expect that these devices will generate a user demand, and eventually focus industry supply into the area of monolithic multipliers, which otherwise has suffered from slow progress over the past five years.

Investigation was also made into the usefulness of the 1-bit serial approach, based on serial-parallel multiplier IC from AMD. Figure 63 shows the latter device is currently the most efficient multiplier type commercially available. The best system design that can be developed to exploit it uses a system timing which alternately reads n bits serially with writing back n bits serially. This timing is a natural result of the double-length

- MOST EFFICIENT OF CURRENTLY-AVAILABLE MULTIPLIER IC's
- PRACTICAL TIMING SCHEME:



- APPLICATIONS: FIXED FUNCTIONS, MINIMUM IC's
- RECOMMEND MORE EFFICIENT VERSIONS FOR HIGH SPEED LOGIC STUDIES, BUT NOT FOR MICRO SIGNAL PROCESSOR

Figure 63 -Serial-Parallel Multipliers

result produced with that IC design, operating between two memories. Only a minimal amount of fixed-function signal processing can be achieved with these serial devices. The net thruput per IC still is not as great as a parallel approach, because of the 2n clock cycles per processed word despite the potential for very high speed clocks. These investigations do, however, provide further bounds on the size-thruput competition for the micro signal processor design.

A fast one-step shifter or shift-barrel is a very desirable chip for a pipelines signal processor. Such a device is available today only in small sizes, such as 4 bits of 0-7 or 0-3 shifts. Hence, we investigated the combination of a decoder with a multiplier chip to produce this function with a minimum number of available chips. Further examination of the power and speed penalties have caused us to reconsider the merits of this concept. Instead we propose to do this function as part of a gate array, or to await the introduction of a commercial IC, as is now planned for the ECL 10800 family.

4.5 Memory

Our micro signal processor design emphasizes replacing registers and selectors wherever possible with scratchpads for cost and component count savings. Table 26 shows the relative cost per bit savings obtained with denser, but slower storage elements.

TABLE 26
MEMORY TRADEOFFS

Storage Type	Packaged Cost Per Bit	System Speed (nsec)
Register-LS TTL	50	25
16 x 4 RAM-LS TTL	8	40
256 x 4 RAM-BIPOLAR	8	70
1024 x 1 RAM-BIPOLAR	4	70
4096 x 1 RAM-MOS	1	400 (cycle time)

The clear trend for memories today is towards higher bit densities, with downgrading of memory hierarchy concepts. Bipolar is today 1K by 1 or 256 by 4 with a just-announced IC of 4k by 1. MOS gives 1K by 4 or 4K by 1, with some 16K memories being sampled. Sizes most likely to emerge with the 16K memories are 16k by 1 and 4k by 4. As memories get denser, the need for a hierarchy of memory speed goes away, saving the overhead required to make this hierarchy invisible to the programmer.

Signal processor memory needs differ from those of GP computers in that read - write cycle time becomes the critical parameter, not the faster access time seen on some memories.

Other memory considerations include dynamic operation, error correction and line-oriented memories. Controlling refresh operations on dynamic memory can consume significant amounts of a small computer's logic, although a signal processor seldom allows data to sit still for long. Error correction is being included in many 4k and 16k memory systems because of pattern sensitivities. Yet signal processor gain and thresholding algorithms can tolerate most errors in data bits. Line or serial type memories, such as CCDs will always have the edge on density, but they necessarily restrict the problem solution to successive processing algorithms or require separate working (RAM) storage. Our initial preference is for simplicity even at cost of lower densities.

The status of programmable elements is presented in Table 27, covering PROM, ROM, PLA, FPLA and EROM. The trend has been toward larger size DIPS to hold more bits. Erasable ROM's have clearly taken much larger packages than the corresponding PROM, aside from the factor of 4 or more in speed. Fast PROM is in the region of 8K today for single source, 4K for multiple sources. ROM's are advocated only when the volume of production and bit densities justify rejection of the PROM option. We foresee a new series of PROMs and FPLA's being developed which will

TABLE 27

PROM/ROM/PLA/FPLA.EROM

NO. BITS/IC	16 PINS	18 PINS	22 PINS	24 PINS	28 PINS
0.25K	32 x 8 PROM				
0.5K					32 x 16 EROM
1K	256 x 4 PROM			256 x 4 EROM	
1.5K				14 x 8 x 48 FPLA	16 x8 x 48 FPLA
2K	512 x 4 PROM			256 x 8 PROM	
3K				14 x 8 x 96 PLA	16 x 8 x 96 PLA
4K	1024 x 4 PROM	1024 x 4 PROM	1024 x 4 EROM	512 x 8 PROM 1024 x 4 EROM	
8K		2048 x 4 PROM		1024 x 8 PROM 1024 x 8 EROM 2048 x 4 EROM	
16K				2048 x 8 ROM	

NOTE: ALL PROM's ABOVE HAVE EQUIVALENT ROM

incorporate register or latch buffering on either the input or the output paths.

Applications for PROM and FPLA in the micro signal processor design are in macro-to-micro decoding, macro program store, tables, and replacement of low density logic. Because of logistic and speed considerations, its use for logic substitution will be minimized.

4.6 Control and Interface

Control decoding will emphasize the use of memory devices. For example, the macro to micro decoding will be PROM, to minimize size. Routing will be done as far as possible by memory address selection rather than identifiable low-density selectors. The logistic headaches of PROM and PLA will receive consideration in this design approach.

Mixing devices from different micro processor families, such as CPU slices of one family with sequencers of another family and interface circuits of a third, allows exploiting the strengths of each different manufacturer's designs, and gives greater performance for less pieces. Yet a potential problem exists of having to support a larger variety of captive lines after the inevitable design shakeout occurs. Technological superiority will not guarantee IC survival against a strong competitor's early market penetration, volume yields and use of established manufacturing processes which are industry wide.

Bus transceivers and bidirectional I/O port chips dominate the interface IC developments. Raytheon's approach of a multiclock macro controlled AU allows bus devices to efficiently share the data paths between micro signal processor elements.

The FIFO is a key part of the proposed control scheme. It allows separating the instruction sequencing operations from the remainder of the processes. MOS FIFO's have been available for

a couple of years now. Two bipolar FIFO's have recently been announced, including a 16 word by 5 bit IC and a 64 word by 4 bit IC. The later is the prefered unit as it is larger and also compatible with the older MOS one and fits the industry preference for 4-bit slices.

The more traditional computer interface devices, such as vector interrupt handlers, and programmable interval timers, are useful only when considering the placement of the micro signal processor into a problem solving system. Use of such specialized, GP computer oriented IC's is confined to the vicinity of the 16 bit GP I/O bus of the micro signal processor, which handles primary mode commands and BITE information. Air Force efforts towards standard interfaces may have the primary influence in this area.

Data addressing control will emphasize RALU slices more than sequencers. Signal processing addressing is normally incremental, but not always increments of +1, +2, or -1.

An item which could simplify interfacing is an "output signal multiplexer". This combines a programmable trigger generator with a multiple output binary rate multiplier. Such an item could easily be made and have wide application, but has not yet received tha attention of the semiconductor industry. Applications include radar action timing, replacement of multiple D/A IC's, phased-array beam steering command generator, netted computer system control, and others. Design and fabrication of th this item appears feasible using Raytheon's 300 gate array capability. Figure 64 shows the interface definitions for this device.

4.7 Selected Building Blocks

A summary of the micro signal processor building blocks expected from commercial LSI is presented in Table 28. Two examples are often given for each category--an immediately available type and a type whose introduction is planned within 6 months to a

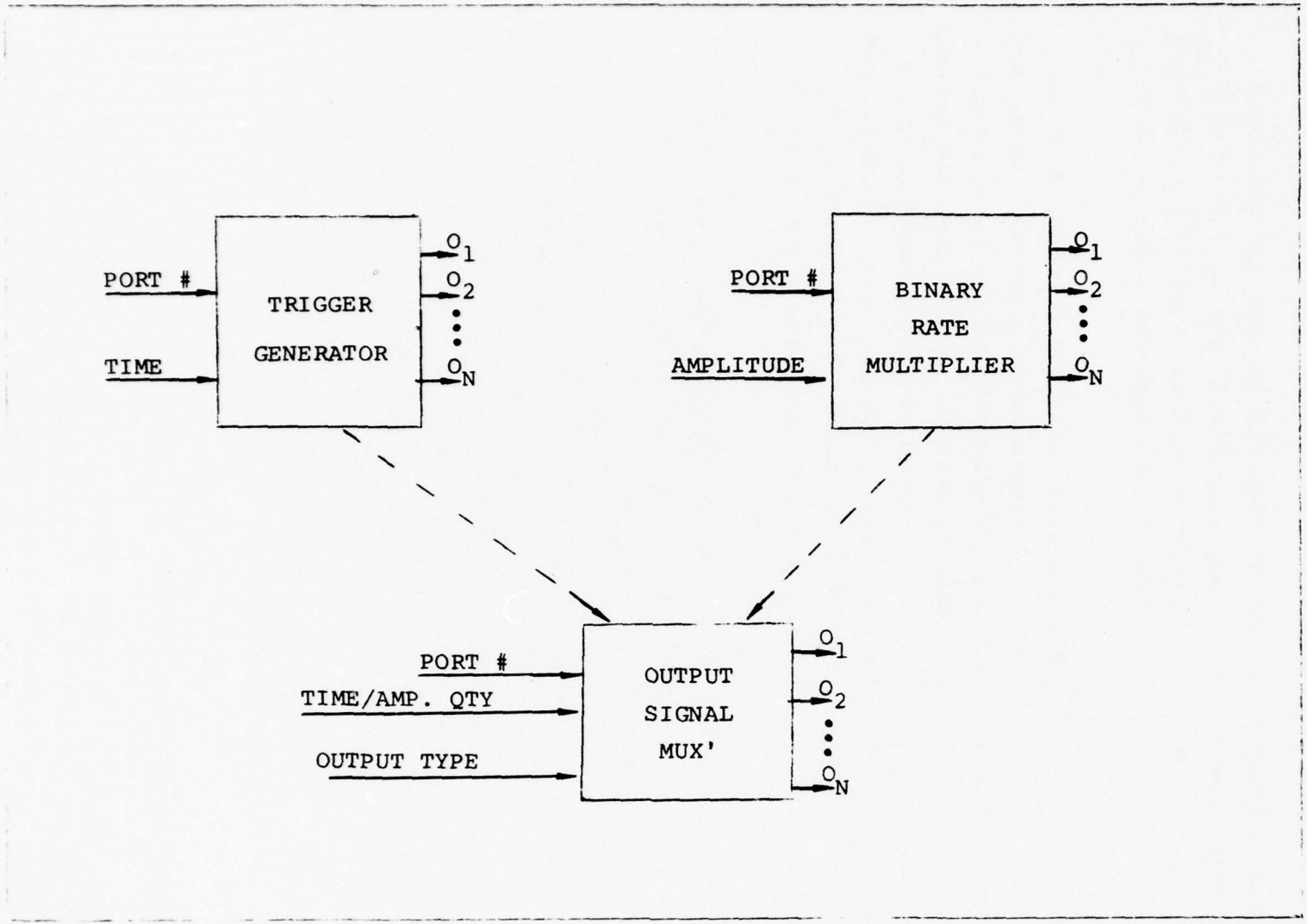


Figure 64 Output Signal Multiplexer Concept

TABLE 28
MICRO SIGNAL PROCESSOR BUILDING BLOCKS

	NOW	LATER?
• SEQUENCER		
• 4 BIT SLICE/20 PINS	AMD 2911	TI 54S482
• 10 BIT UNIT/40 PINS	SIG 8X02	FCD 9408
• RALU		
• 4 BIT μP SLICE/16 WORDS	AMD 2901	
• 4 BIT PIPELINE SLICE	MMI 6702	
• BUFFER RAM		
• 16 x 4 DUAL OUTPUT	AMD 29705	
• 64 x 4 FIFO	MMI 6741	
• MULTIPLIER; DIVIDER		
• 8 x 8 COMBINATORIAL		MMI 6755
• BUS 16 BIT MULT/DIV		MMI 6750
• VECTORED PRIORITY INTERRUPT		
• 8 INPUTS EXPANDABLE	INTEL 8259	AMD 2914
• RAM		
• 256 x 4/TS OE/DATA IN ≠ OUT	FCD	RCA CMOS/SOS
• 1K x 1 FAST	FCD	
• 1K x 4 SLOW STATIC	AMD, INTEL	
• 4K x 1 FAST STATIC		FCD, TI
• PROM		
• 1K x 4 IN 18 PIN DIP	MULTI-SOURCE	
• 2K x 4 IN 18 PIN DIP		SIG
• MOS μP's	8080, 6800, F8	EA9002, Z80
DYNAMIC RAM - 16K		
• FPLA		
• 16 x 8 x 48/50 nsec	MULTI	
• MISC		
• 3 TO 8 DECODER	AMD 25LS 2538	

year.

Two types of sequencer are included, a four-bit slice and a ten-bit slice. Further machine design will determine if both types are necessary.

Two types of Register-Arithmetic-Logic-Units are included, the conventional micro processor slice and the pipelined equivalent. A more optimized pipelined bit slice units could be made with a mixture of 300 gate arrays and buffer RAM's such as the multi-port 16 word by 4 bit unit.

FIFO's are desired to mesh variable execution time sequencing instructions with the fixed execution timing of the arithmetic pipeline. They are also useful for buffering peripheral I/O data into a larger block of words which is more manageable in a software scheduling sense.

Multipliers are an important part of any signal processor. Thus, we shall assume availability of at least a fast 8 by 8 multiplier, with preference towards larger sizes, such a 8 by 12 or 8 by 16, operating within a pipelined 100 ns clock cycle. For handling infrequent division operations, the one chip multi-cycle approach postulated for the MMI7650 is more desirable than sending data back to the driving GP computer or performing multiple passes through the signal processor's arithmetic unit with a "divide by way of multiply" algorithm.

Vectored priority interrupts is a function which we shall shape around available designs, since interrupts in a μSP need not be handled extremely efficiently or fast.

Random access memory availability includes not only the speed-density trade-offs, but also the width of the data path of one chip, and the coupling or independence of the data input and output paths. The 4K words by 1 bit fast static memory is

expected to dominate fast data needs eventually, although the 1K by 4 bits is useful in the control memory area. Bulk data memory requirements should use random access memory of 16K densities or higher, even if that requires dymanic storage and refresh mechanisms.

MOS micro processors can be categorized into: a) current leaders, such as the 8080 (for the most people adding parts that meet with it), the 6800 (for technical niceties) and the F8 (for absolute minimum number of IC's) and b) the improved micro processors like the EA9002 and the Z80.

Other IC's to which some, but not much, attention should be paid include the FPLA and the 3 to 8 decoder. The latter is useful in block floating point operations for storing exponents in the data stream, as well as for traditional selection tasks. The FPLA may have some applications in the scaling element, but is much weaker in capability than Raytheon's 300 gate array.

SECTION V
SOFTWARE

5.1 Simulation Objective

The simulation activity for the micro signal processor is based on a mixed functional and register level model of the architecture. The result is useful in both design verification and software debug. Under this study the assembler and simulators for the address generator and the sequencer was developed and coupled. Provisions were made for easy extension to the whole μSP under the next phases of this study.

5.2 Philosophy

The simulation is designed to run on a CDC Cyber 73. It is assumed that Fortran IV had been chosen as a standard to ensure the transferability of the simulator. PMS (Processor, Memory Switch) and ISP (Instruction Set Processor) descriptions are employed to be certain that modelling accurately tracks the computer architecture. It is assumed that the FIFO between the Sequencer and ADGEN (Address Generation) functions acts as an exclusive asynchronous communication buffer i.e. the sequencer, and ADGEN portions can be treated as independent processors with the exception of the I/O protocal via the FIFO. In terms of the control structure both processors are slaved to the state of the FIFO.

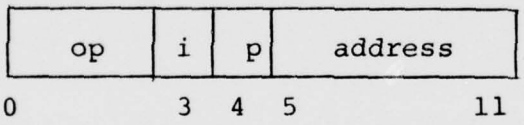
Effort was focused on the control element first rather than on the pipeline. Pipeline sumulation is straight forward because of the fixed clock count nature of all macro execution. A functional level simulation is most appropriate, which in turn depends on defining a large collection of macro instructions. More insight and confidence into the operation of this μSP approach is gained from developing the control simulation first.

The simulation can be viewed as an assembler/simulator since a portion of the simulator will be an assembler. Two uses of the system can be envisioned and are included in the design philosophy: 1) design aid, and 2) trainer. It's use as a design aid is obvious. It's use as a trainer resides in the fact that it is an assembler/simulator with which a user can practice coding prior to equipment availibility.

The software architecture is rather classical. It should be noted that the models themselves form a small portion of the system whereas the rest support user I/O and system initialization functions. This type of structure permits easy modification of either the model or user portions independently.

All logic levels in this simulation effort are described as 0 or 1 (ie. low or high) rather than true or false with the exception of TC-test condition - which will always be true=low.

The following bit numbering convention is used:
the leftmost bit position will be labelled bit 0, with bit lables increasing monotonically to the rightmost bit-e.g.



Negative numbers are represented in two's complement form within the μSP. There are two ways in which negative numbers can occur. First, the programmer may insert them into a register from the quantity field by a Sequencer-LQ instruction, and second, repeated execution of the Sequencer-DR instruction will eventually result in a negative value.

To facilitate recognizing and manipulating negative numbers in two's complement form, some non-standard Fortran functions have been used, (i.e., MASK and SHIFT, and user defined functions built up from these). These are the same functions which are used to decide instructions and to implement the ISP concatenat function,

as described in reference 2. It has previously been decided that a simulated μSP word, (of whatever length), will occupy a full CDC word. Hence, to take advantage of CDC defined relational operators and arithmetic, the sign bit of the μSP word must be propagated to the left to fill the entire 60-bit word. Once this has been done, two's complement numbers can be converted to one's complement, any arithmetic function may be performed and the result converted back to its two's complement representation.

5.3 Orientation

The Micro SP (signal processor) architecture can simply be viewed as two dissimilar programmable processors linked by a FIFO buffer acting as a one way asynchornous communication buffer (see below).

The two processors are referred to as the Sequencer and the Datapipe. The Datapipe exercises control over data memories and an arithmetic pipeline. As such it is the "number cruncher" portion of the SP. The Sequencer exercises control over the Datapipe by means of pointers that are converted and used as program counters within the Datapipe.

The Sequencer is master with respect to the Datapipe and transfers its Datapipe pointers via the FIFO. The FIFO is required to compensate for the asynchronous operation of the two units. When the FIFO is full the Sequencer clock is inhibited. When the FIFO is empty the Datapipe clock is inhibited.

The Datapipe is composed of two major parts i.e., the ADRGEN (Address Generator) which controls data flow and the arithmetic pipeline which processes the data. Both of these units

are separately programmable.

The Sequencer code, which is not selfmodifiable, resides in Sequencer Memory. The instruction is composed to two major segments i.e., Sequencer control and Datapipe control. The Sequencer control portion of the instruction effects the next address control for the Sequencer PC (Program Counter). The Datapipe control portion of the instruction is loaded (subject to test criteria) into the FIFO. As time is available, the Datapipe reads the Datapipe control data from the FIFO and distributes the two pointers contained in it to the ADRGEN and pipeline respectively.

Figure 65 illustrates the data flow. The figure is highly simplified and is presented in a way that would be most tutorial for the assembler effort.

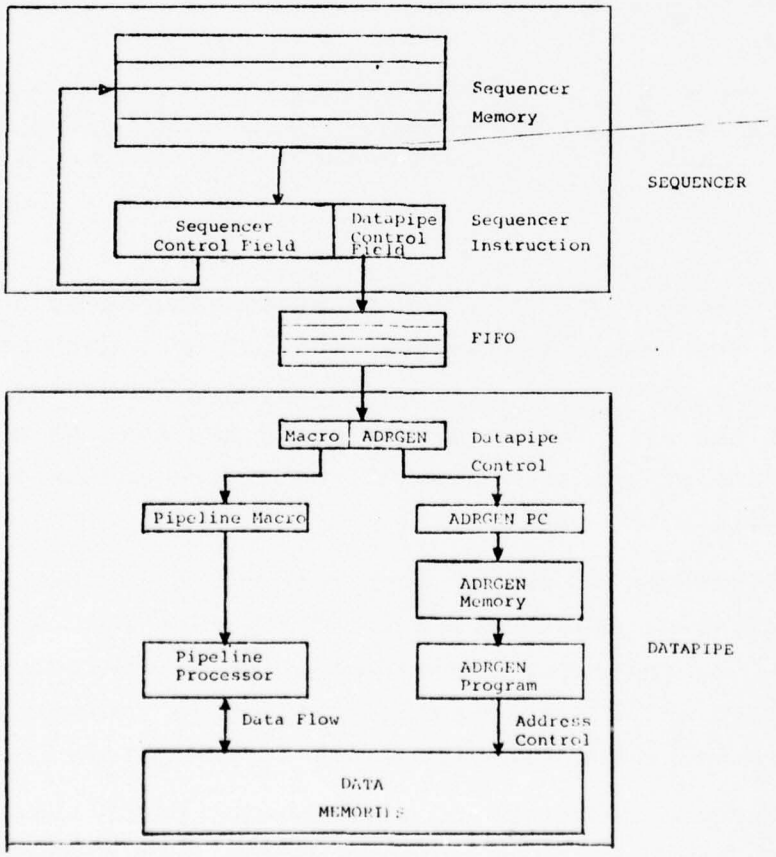


Figure 65 - μSP Instruction Data Flow

Although code for the two processors is independently executable the two are not independent from the user viewpoint. Since the user expects to refer to ADRGEN and macro (i.e., pipeline) programs by name rather than address the assembler must associate the two via its symbol table. Implementing this process implies that symbolic entry points must be defined for the Datapipe. From the user standpoint the most advantageous way of doing this is to be able to freely intersperse the code for the two processors into a merged but logically coherent source file. The task of the assembler then would be to separate the two files, assemble the ADRGEN code, then assemble the Sequencer code.

The macro (pipeline) pointer values would have to be assigned by the user in the Sequencer program. For simulation purposes, however, the pipeline will be modelled with its functions included i.e., no assembler requiremente exist for the pipeline at this time.

5.4 Micro Signal Processor Simulation

The purpose of the micro signal processor simulation is to simulate the execution of the SP resident software. This sumulation is a mixed functional and register level simulation of the micro signal processor architecture with features that enhance its adaptability for reconfiguration purposes. The simulation is coded in FORTRAN and is executable on the Cyber 73 at Raytheon in Bedford.

In support of the task two new assemblers were developed: one for the sequencer, and the other for the address generator portion for the SP. These assemblers are configured in such a way that they are equally useful for developing object code for either the simulator or for actual hardware. These two mutually independent symbolic cross assemblers permit the user to specify the source code in a format that, by field, closely parallels that of the generated object code.

The simulation portion of this assembler/simulator simulation system accesses the object code files, creates instruction memory images, and simulates the instruction execution of the two instruction processors (sequencer and address generator) subject to the effect of the FIFO interface that serves as an asynchronous interface buffer between the two devices. The user exercises control over the simulation by means of a user run stream that includes trace and debug features. They permit the user to 1) control the state of the machine (e.g., halt, run), 2) write into or read from instruction memories, and 3) trace critical registers during program execution.

5.5 Program Organization

The micro signal processor simulation consists of three separate programs that operate under the control of the Cyber 73 NOS operating system. The three programs are: the sequencer assembler, the ADRGEN (address generator) assembler, and the simulator (sequencer/FIFO/ADRGEN/ instruction set processor).

The sequencer assembler accepts source code (as shown-in Figure 65 and generates object code (identical in format to that for the sequencer instruction memory) which is placed on a CDC file. The ADRGEN assembler accepts source code (see Figure 65) and generates object code (identical in format to that for the address generator memory) which is also places on a CDC file. The two files are input files to the simulation program. The output of the simulator is a description of critical register states.

Figure 66 illustrates the simulation system flow. Figure 67 shows the hierarchy diagram for the simulator itself.

The μSP simulation system requires the sequential operation of three programs; the ADRGEN assembler, the sequencer assembler, and the simulator. There is no co-residency requirement. Only the output file of the assemblers need be preserved for the simulator.

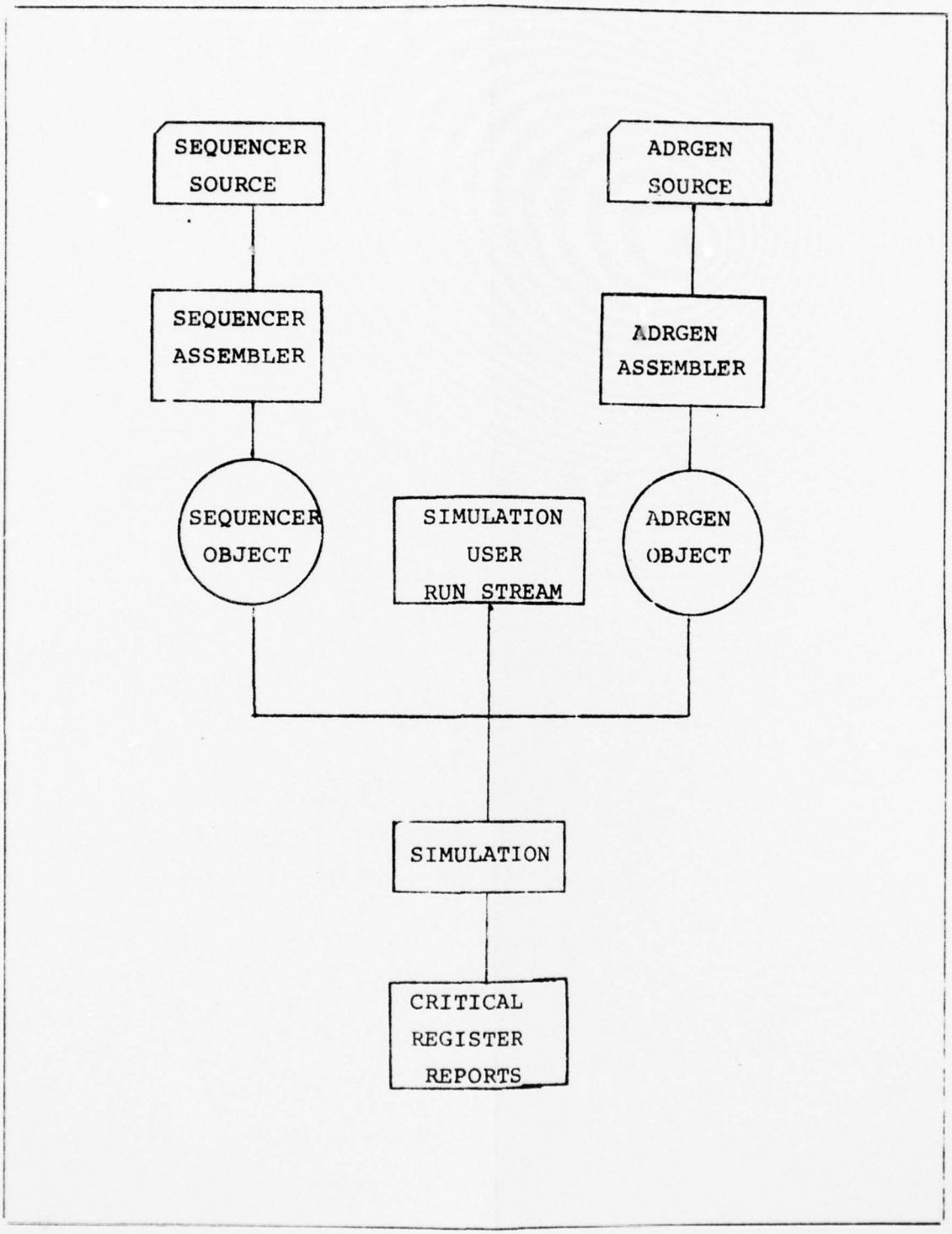
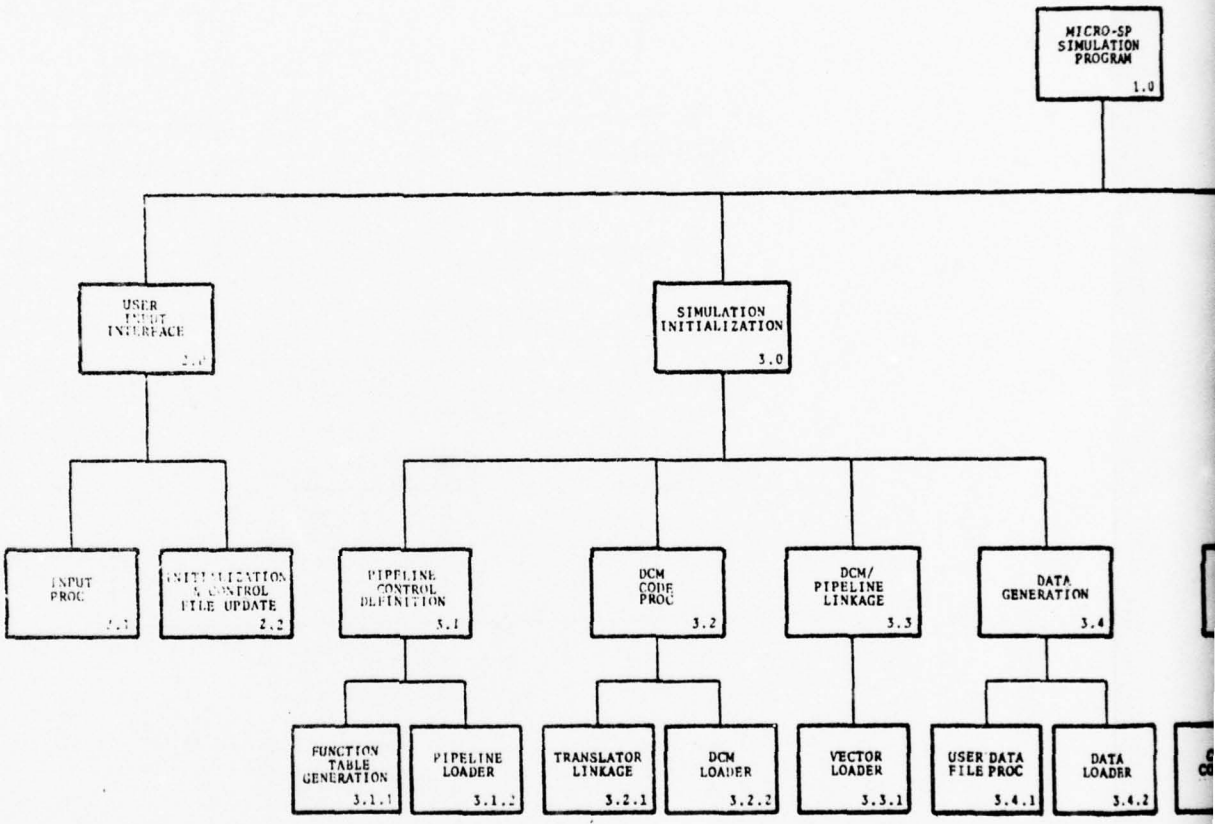


Figure 66 - MicroμSP Simulation System Flow Diagram


MICRO-SP SIMULATION PROGRAM 1.0
USER INPUT INTERFACE
SIMULATION INITIALIZATION 3.0
INPUT PROC
INITIALIZATION & CONTROL FILE UPDATE 2.2
PIPELINE CONTROL DEFINITION 3.1
DCM CODE PROC 3.2
DCM/ PIPELINE LINKAGE 3.3
DATA GENERATION 3.4
FUNCTION TABLE GENERATION 3.1.1
PIPELINE LOADER 3.1.2
TRANSLATOR LINKAGE 3.2.1
DCM LOADER 3.2.2
VECTOR LOADER 3.3.1
USER DATA FILE PROC 3.4.1
DATA LOADER 3.4.2

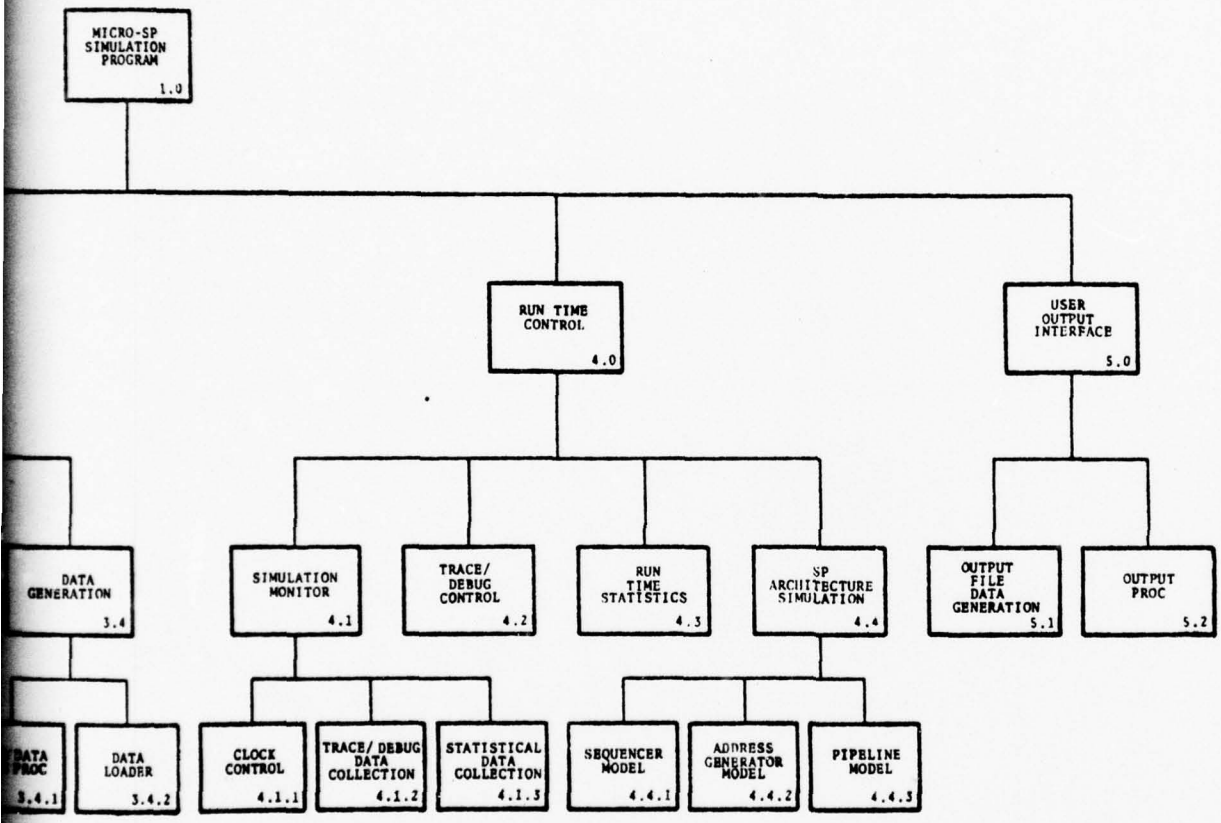


Figure 67 - Hierarchy Diagram

5.6 Software Development Status

The software for the micro signal processor simulation was developed using Bedford Laboratories procedures for software development. The requirements for the simulation were specified in Instruction Set Processor (ISP) format for clarity. These descriptions spanned the sequencer, FIFO, and address generator portion of the system. They did not cover the pipeline functions or structures.

The simulation spans events from source assembly to MAR outputs. The pipeline functions are not incorporated.. The trace/debug functions are incomplete as are the data reduction functions.

Extended FORTRAN features of the Cyber 73 system were used to facilitate code development. The areas of such usage are well defined and subsequently may be subjected to standardization.

Testing of all programs was conducted as generated. A final acceptance test consisted of the assembly and execution of an FFT (Fast Fourier Transform) program which is described later. The results of this benchmark were verified against anticipated register states (e.g., memory address registers) and found to be consistent with hardware design.

After the programs had been completely checked out the program configuration control items, deck and listings, were placed in the Digital Systems Laboratory Program Library. Final documation consists of a simulation user's manual - Raytheon BR-9632. These specifications contain all of the information necessary for the use of the complete program. A copy of the program listing is enclosed as an appendix to that report.

5.7 Future Development

The design of the simulator includes features which anticipate the future incorporation of trace/debug, statistics, and report generation options. Another improvement is to link the two assem-

blers so that sequence instruction can refer to ADRGEN entry points by name. The simulator currently has no arithmetic capability at the pipeline level. This should be done to prove out the correctness of macro programs (in an arithmetic sense). Data can be externally generated by a user program and loaded via a data memory loader. This feature would be a desirable adjunct to pipeline testing. The trace/debug features permit a wide variety of via time checkout of intermediate data. The statistical data reduction would require the implementation of the data collection and reduction function library. The report generation could be as elaborate as utility dictates.

SECTION VI
CIRCUIT TECHNOLOGY

6.1 Logic Technology Choices

The status of today's contending circuit technologies was surveyed to aid in choosing the implementation directions for this μSP for up to the next five years. This review is based primarily on information gleaned from the public literature. Although this does not yield the latest, hottest things in progress in the private corners of semiconductor laboratories, it is within the self-interest of makers of the latest circuit technology to drop hints as to their direction and expectations. Moreover, due to the high capitalization factor of the semiconductor industry there is a strong and almost overwhelming impetus to push the process presently being worked to its limits before considering the jump to a drastically different process.

A strong bandwagon effect exists to follow the mainstream even if it is not as good in certain aspects as other logic technologies, just because everyone else does it that way. Reasons for this "follow the leader" tendency include customer acceptance, the need for background products to pay for new developments, reliance on the same information sources, and strong mobility of professionals between companies, just to mention a few.

Considerable filtering was done on these public sources of information to try to separate the truth from market testing and puffery. Still, as an industry, we talk a lot about how good we do, including freely publicizing the latest chip masks and process outlines. This is probably because such things are no secret anyway once the first chip gets sold.

Figure 68 and 69 present the data for this circuit review. Note the distinction between approaches in production and those in development. The fact that a new process is in production

	STTL	100K ECL	Si GATE CMOS	LSTTL	EFL	DEPLETION LOAD NMOS	DEPLETION LOAD DMOS
SPEED (ns/gate)	2	.75	10	3.5	10	12	3
POWER (mW/gate)	20	10	.05	2-3	1.7	3	.7
POWER-DELAY (pJ)	40	5-30	.5	7-10	17	38	2
GATES/mm^2	40	40	45	50	50-75	100	140
MANUFACTURERS	TI	FAIRCHILD	INTEL INTERSIL	FAIRCHILD's MACROLOGIC	TRW	ZILOG ELEC. ARRAYS	NIPPON ELEC.
STATUS	IN VOLUME PRODUCTION	IN VOLUME PRODUCTION	IN VOLUME PRODUCTION	IN VOLUME PRODUCTION	IN DEVELOPMENT	IN PRODUCTION	IN DEVELOPMENT

Figure 68 - Logic Technology Survey I

	CMOS/SOS	CMOS	DMOS	VMOS	DIGITAL CCD	IIL	Si GATE NMOS	S-IIL
SPEED (ns/gate)	1-4	50	5	5	40	10-50	10-50	10
POWER (mW/gate)	.07-.3	.1	4	4	.1	.01-.1	.2-1.	.003
POWER-DELAY (pJ)	.25	4	20	20	4	.1-1.	10	.03
GATES/mm^2	150	200	225	225	300	120-320	500	800
MANUFACTURERS	HEWLETT-PACKARD RAYTHEON RCA	MOTOROLA RCA	SIGNETICS	AMI	TRW	FAIRCHILD TI Signetics	MOSTEK INTEL	PLESSEY
STATUS	IN PRODUCTION and in DEVELOPMENT	IN VOLUME PRODUCTION	IN DEVELOPMENT	IN DEVELOPMENT	IN DEVELOPMENT	IN PRODUCTION	IN VOLUME PRODUCTION	EXPERIMENTAL

Figure 69 - Logic Technology Survey II

doesn't necessarily mean that it's technical value is obsolete when looking forward over the next several years. Higher resolution lithography techniques alone are likely to add years, or more service to today's "workhorses".

A key parameter is the packing density in gate/mm^2. The spread in packing density between what is in high volume production and what is experimental is about an order of magnitude. Not all that density improvement can be realized due to chip partitioning problems and I/O requirements. Thus, those μSP element design parts considered for single chip implementation should be subject to a marketing analysis of risk, expense, and payoff. The payoff which justifies pursuing a higher packing density must be significant improvements in parameters such as power, speed, chip totals and/or fabrication ease.

Some general statements can be made about the most useful aspects of each contending circuit technology. LSTTL is well known, has a good speed-power product and is the industry workhorse. I^2L will provide considerably higher density, at slower speed, and probably with TTL interface levels most of the time. CMOS-SOS provides radiation hardness and low power for equivalent speeds of LSTTL. ECL gives at least a factor of two more speed than TTL, although the higher power densities require special cooling considerations. NMOS is moving up in speed, with variations like DMOS providing even better process control

We conclude that with such a choice available, no one technology is absolutly superior. A few complementry technological points should therefore be chosen to cover the full range of needs. For example, to encourage immediate applications of the μSP a version based on available LSI building blocks, namely LSTTL, is envisioned. For hybrid packaging, where power limitations may be more significant than chip count, CMOS-SOS is the most viable candidate. Both will use clock cycles in the 100 - 150 ns region,

depending on desired operating voltages and temperature margins. An ECL type version has some attractiveness for the future as applications push toward greater performance needs. An ECL version, however, will probably use twice as many parts as a CMOS/SOS version, assuming the present technology trends continue.

6.2 Gate Arrays

Considerable interest has been generated recently in gate array approaches to LSI. Among the latest rumors are a 2000 gate I^2L array being postulated by Signetics. Consequently we have made a survey of companies mentioned in public literature as having a gate array capability. The results of this survey are presented in Figure 70 and 71.

We draw significant insight from hands-on experience at out Microelectronic Facility. There, high speed gate arrrays were developed which closely follow the packaging density, integration level and reliability of state-of-the-art custom LSI circuits, but which can be personalized on the final interconnect levels at low development costs. Such arrays can be manufactured economically in very small production quantities of each personalizations, have short one month development cycle, and have very predictable performance. They can replace all the logic in a system with personalized LSI, but are especially desirable when mixed with the best off-the-shelf LSI devices to iliminate any SSI or MSI logic. Arrays based on Schottky TTL circuits have been developed with different speeds and power dissipation levels, all providing high speed and good drive over the military temperature range. Complexity has increased from 24 gates per array in 1968 to the present 300 gates with 5 nsec delay, with 100 gate arrays in advanced development. Recently a 1600 transistor CMOS/SOS array has been developed, having faster speed and lower power than the TTL version. A 5000 transistor array is under development, giving roughly 1000 gate capability.

MANUFACTURER	PRODUCT	TECHNOLOGY	DESCRIPTION (gates/array)	SPEED (ns/gate)	POWER (mW/array at Speed)	PINS (leads/array)
HARRIS	AOP (for NASA)	TTL	116			40
EXAR	XR-400	I^2L	256			16
INTERDESIGN		NMOS	262		39	8,14,16,18,24,40
STEWART-WARNER	SWAP	I^2L	208	75	375	16
	SWAP	I^2L	408	75	750	24
TRW		TTL	158	40	100	40
RCA	MABB	Si-gate	560	40	50	40
	TCC 040	CMOS	168	40	400	40
	TCC 051	CMOS	276	40	600	48
	TCC 220	C^2L (Concentric CMOS)	168	20		40
	TCC 221	C^2L	276	20		48
	TCC 222	C^2L	410	20		48
	TCS 090	CMOS/SOS	182	5		40
	TCS 091	CMOS/SOS	300	5		48

Figure 70 - Gate Array Survay I

MANUFACTURER	PRODUCT	TECHNOLOGY	DESCRIPTION	SPEED	POWER	PINS
INTERNATIONAL MICROCIRCUITS	XXL	CMOS	800	30	400	24,28,40
	XL	CMOS	400			24,28,40
	L	CMOS	200			16,18,24,28,40
	M	CMOS	100			14,16,18,24,28
	S	CMOS	50			14,16,18
MOTOROLA	XC 160	S-TRL	160	25	480	14,16,24,40,42
	XC 400	S-TRL	400	25	1200	
MICROCIRCUIT TECHNOLOGY	707 MAS-TERSHIP	NMOS		10	.26mW/gate	16,18,24,28
RAYTHEON		S-TTL	300	6	1500	64
		CMOS/SOS	300	5	300	40
HUGHES	ULG	ECL	120	2.5	4800	

Figure 71 - Gate Array Survey II

AD-A047 997 RAYTHEON CO BEDFORD MASS MISSILE SYSTEMS DIV F/G 17/9
HIGH SPEED MICRO SIGNAL PROCESSOR STUDY.(U)
SEP 77 G N SHAPIRO F33615-76-C-1339
UNCLASSIFIED AFAL-TR-77-52 NL

3 OF 3
AD A047997

END
DATE FILMED
1-78
DDC

The two key concepts that make this array approach a viable candidate for μSP logic, and not a technology plaything, are computed aided design (CAD) for mask generation and compatibility of the array fabrication with an outside manufacturer's STTL or CMOS/SOS line. The CAD system is particularly cost-effective because experience with the layout of over 200 complex MSI/LSI devices and over 100 hybrid circuits was shown that high yields can be achieved. Such yields allow confident estimates of hardware implementation and check-out time.

We conclude that the most worthwhile gate array candidates today are the CMOS/SOS and for higher speed at lower density, some variation of ECL. I^2L is a likely candidate for future years, in that much greater density may be achieved at some speed loss.

6.3 High Density Packaging

A fundamental packaging problem is the 1 $in.^2$ occupied by a 16 pin DIP costing between $0.5 to more than $10.0 as a function of the number of printed circuit layers and production volume. Partial solutions include:

- Motorola's QUAD-IN-LINE package cuts 40 to 64 pin DIPS to almost half card area
- Flatpacks take 1/2 to 2/3 volume of DIPS, but are not the current industry standard
- Increased signal multiplexing keep pin totals to small DIPS but also increases design complexity

The most promising approach in our opinion is using chips on thick film ceramic substrates, e.g., hybrid packaging. For comparable production volumes and complexities to the above printed circuit approach, a multilayer ceramic substrate cost varies from 2 to 50 dollars per square inch. Now component mounting densities are limited primarily by power dissipation. Thus for a reasonable mix of MOS and bipolar devices, a 20 components/$in.^2$ is plausible. The cost for mounting each device in a hybrid is

then between $0.10 and $2.50, an order of magnitude price improvement.

Total system costs can thus be drastically lower with the intelligent use of hybrid packaging. The components are purchased properly tested, but without extra packaging, and are thus inherently less expensive than flatpacks of DIPs. The key savings are in the reduced number of modules required to do the total job because of the higher packaging density achieved.

Reliability should be significantly greater with hybrid packaging. The large number of solder connections from signal chip carriers (eg. DIP or flatpack) to printed circuit board are replaced by a very much smaller number of connections from hybrid substrate to circuit board.

Figure 72 illustrates the expected size and production cost tradeoffs. Packaging approaches vary from the extreme of all commercial DIPs to the most promising combination of personalized gate arrays with commercial LSI together on ceramic substrated (hybrids).

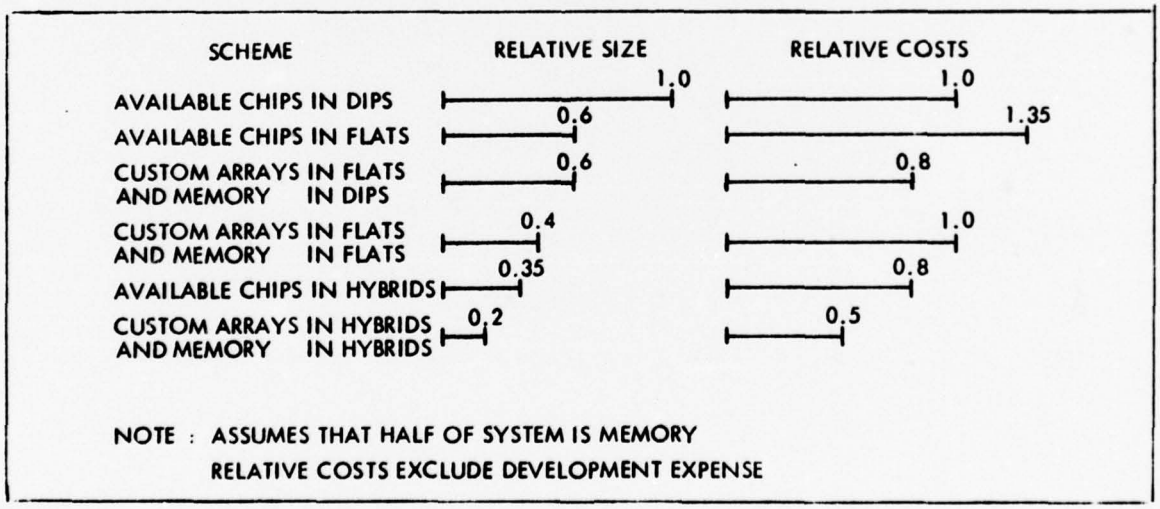


Figure 72 - System Packaging Size and Cost

The tradeoff thus exists between pushing density on one chip verse multi-chip hybrid packaging. The latter involves lower risk than pushing chip technology, but the former has historically had greater payoff.

SECTION VII
DEVELOPMENT PLAN

7.1 Background

The High speed Micro Signal Processor study resulted in a definition of general specifications for a set of micro signal processor functional elements. These elements form a modular, expandable basis for a spectrum of avionics signal processor configurations. The study was concerned with architectural planning and functional partitioning to maximize the use of mainstream commercial microprocessor devices and/or LSI array type building blocks without restricting the design implementation. A functional level simulator design was initiated and partially completed. Documentation includes an Instruction Set Processor (ISP) definition of the modelled elements along with the code and user manuals.

7.2 Objective

The micro signal processor (μSP) development plan is designed to verify the μSP baseline design and evaluate the advanced device development to support and implement the design. These objectives shall be attained in discrete measurable steps. A low risk processor implementation to establish benchmark data and provide an advanced device test fixture; device development program to develop chips not expected to be available in commercial markets; full processor implementation in advanced technology; and verification of device development program. The program has stressed technology independence in implementation. Approaches that maximize this philosophy are desirable features and will be viewed favorably by the evaluators.

7.3 Program Discussion

The contractor's proposal fulfilling the objectives of the

development plan must provide AFAL with a usable product at each phase of subsequent development, as outlined in figure 73.

The initial step is a technology verification model to provide AFAL with three capabilities:

1. A desk-top micro signal processor which can interface into commercial μP's or existing AF base computing facilities.
2. A test fixture and development tool to test and validate subsequent chip development capabilities in subsystem use.
3. A firmware/software development tool to develop and test algorithms and develop software for the advanced technology model.

The second step is the chip development of critical functions. These chips will be functionally compatible with the technology verification model and can replace the commercial functions allowing AFAL to measure the advanced device performance.

The third step is the development of the remaining chips required to complete the advanced technology model. When each chip is developed, it can be placed into the commercial version to demonstrate the viability of each chip.

7.4 Statement of Work

The requirements to finalize design, fabricate and test the High Speed μSP consists of three phases. Task descriptions by phase are shown below; schedules for each phase is shown in figure 74. The contractor is to provide personnel, materials, and facilities with the objectives to complete the following development and demonstration tasks. (Outlined in figure 75).

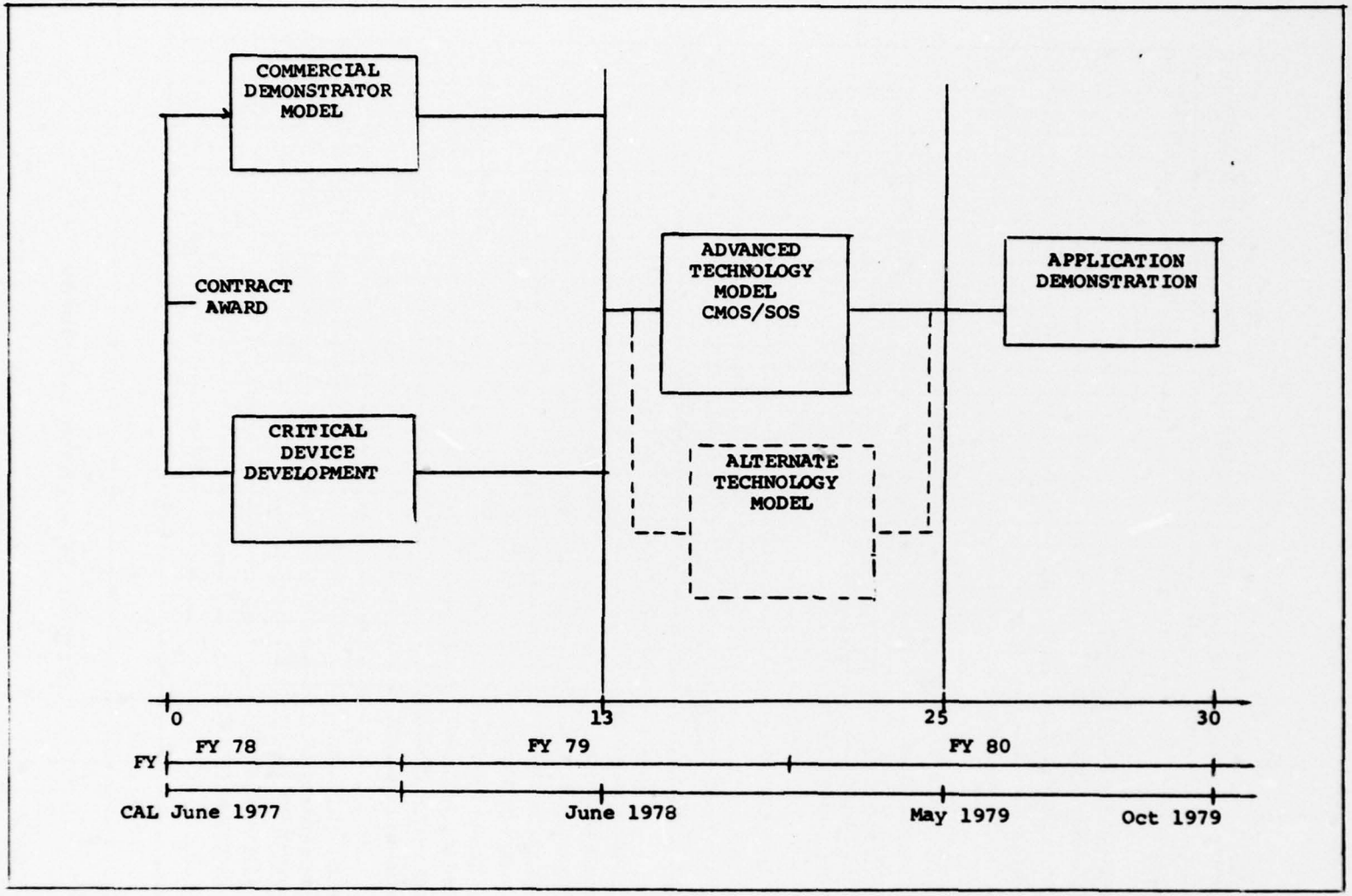


Figure 73 - Development Plan

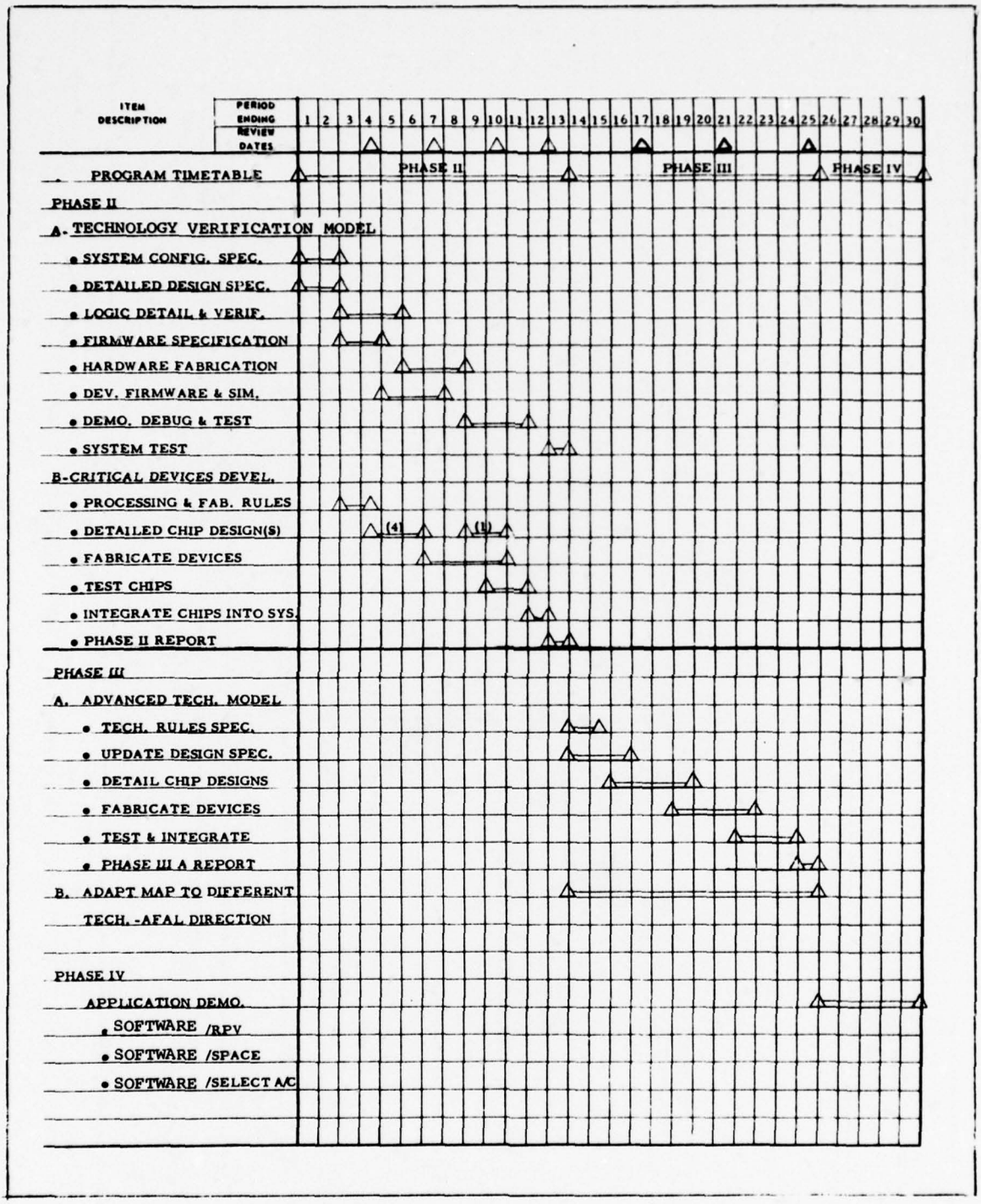


Figure 74 - Phase Schedules

OBJECTIVE: DEVELOP A MICRO SIGNAL PROCESSOR TO SUPPORT MISSIONS FOR MANNED AIRCRAFT, REMOTE PILOTED VEHICLES, AND SPECIAL DEVICE (SPACE) APPLICATIONS. DEVELOP μSP AS A BUILDING BLOCK ELEMENT TO SMALL AND LARGE SCALE PROCESSING CAPABILITY.

GOALS:
1. MAXIMIZE USE OF COMMERCIAL μP FUNCTIONS
2. EVOLVE INTO ADVANCED AF SUPPORTED TECHNOLOGY
3. PROVIDE SPO'S/PROGRAM OFFICES WITH HARDWARE - AFAL DEVELOP MISSION FIRMWARE

PROGRAM:
- ϕ I - STUDY AND GENERAL ARCHITECTURE DEFINITION
- ϕ IIA - COMMERCIAL DEMONSTRATOR - PROVES BUILDING BLOCK DESIGN
  - B - CRITICAL DEVICE DEVELOPMENT - EVOLVING TECHNOLOGY INDEPENDENCE
- ϕ IIIA - ADVANCED TECHNOLOGY VERSION - OPTIMIZE SPEED/POWER AND DENSITY FACTORS
  - B - ALTERNATE TECHNOLOGY VERSION - OPTIMIZE SPEED?, DENSITY?
- ϕ IV - APPLICATIONS IMPLEMENTATION - AFAL/SPO JOINT TEST PROGRAMS

Figure 75 - High Speed μSP Development Plan

7.5 Phase IIa - Technology Verification Model (TVM)

A detailed design shall be implemented using currently available commercial μP elements. The μSP logic design should be verified by simulation and the hardware implementation supported with suitable firmware programs to support subsequent development and testing.

1) The contractor shall provide detailed design specifications for implementing the μSP conforming to the requirements definition outlined in section one.

2) The design specification shall include a specification outlining the requirements for firmware. The contractor shall demonstrate, in appropriate sequences, the capability to process the following algorithms for test purposes.

FFT with input weights
FIR Filter
IIR Filter
Magnitude & CFAR Thresholding
EW Signal Sort
2 Dimensional Correletion

This specification shall be the subject of a design review at the end of month three.

3) The contractor shall implement the approved design specification using CAD systems technology and verifying the designs with the previously developed simulation programs (descriptions provided with the RFP). This stage shall verify that the processor detailed design performs as required; when such verification is established, the contractor will fabricate the technology verification model.

4) The fabricated models will be tested using the firmware and test procedures developed in the design specification. At the conclusion of the development tests, an official acceptance test will be conducted at AFAL facilities in Dayton, OH.

5) The TVM shall be used to evaluate and demonstrate the viability of subsequent chip developments in Phase IIb and Phase III. When these chips are developed, they will be substituted into the TVM, replacing the commercial logic, and run with the previously developed algorithms.

6) A Phase IIa report shall be prepared and will include the specifications, the detailed design data, block diagrams, parts list, and description of firmware and procedures for developing subsequent software algorithms.

7.6 Phase IIb - Critical Device Development (CDD)

These μSP functional elements should be implemented in the desired technology and subsequently integrated into the μSP point design replacing its commercial counterpart. This test will provide incremental verification that the LSI implementation is functional identical to the commercial design.

Critical Devices will be built, superseding equivalent commercial logic in the TVM. Estimated cost are shown in figure 76.

1) The contractor shall design, build, and test 4 chips from the list of candidate critical devices shown in section 1. In the course of this development, the contractor shall develop a functional design specification for this device including device test and

A - COMMERCIAL DEMONSTRATOR (2 UNITS)

- PARTS
- FABRICATE BOARDS
- COMMERCIAL PACKAGE
- ENG/FIRMWARE/MGMT, ETC

B - CRITICAL DEVICE DEVELOPMENT

- 5 CHIP DESIGNS
- ENG SUPPORT/TEST

Figure 76 - ΦII - Processor/Device Development

evaluation data.

2) The contractor shall implement a detailed design of the chips upon approval of the design specification by AFAL. Array technology is recommended for this implementation; to enhance the probability of evaluating the technology design in the TVM.

3) The contractor shall demonstrate the critical device capability by substituting these devices into the TVM and running the baseline algorithms. Relative benchmark comparisons shall be normalized to compensate for attenuation by miscellaneous logic needed to make the TVM interface.

4) A Phase IIb final report will be prepared outlining the conclusions, tast data evaluation, chip designs, and other appropriate technical data.

7.7 Phase III - μSP Advanced Technology Model (ATM)

The remaining functional elements will be designed and verified in a similar fashion as in the TVM phase. Supporting device characterization tests will also be performed to fully describe the devices.

The μSP functional units consist of arithmetic, control and memory elements.

1) The contractor shall identify the remaining devices necessary to complete the elements of an advanced technology model. This list of devices shall include the list provided in Task II so AFAL may evaluate the priority of implementation.

2) The contractor shall design, build, and test a com-

plete ATM using the selected advanced technology, using an AFAL approved specification, in either customer or gate array implementation. The contractor shall demonstrate how he intends to complete customer designs within the timetable available for this phase.

3) The contractor shall develop a test/integration procedure for testing the chips as devices and system elements. The procedure shall allow for substituting these functional elements into the TVM and measuring the performance. The procedure shall insure that sufficient combinations of the functional elements are tested in the TVM before the μSP ATM is tested as a standalone unit.

4) Upon completion of the hardware, system tests using the previously developed algorithm shall be conducted to measure the relative increase in performance in this phase.

7.8 Phase IV - Application Verification Phase (AVP)

Application firmware developed by AFAL/Contractor will be used to demonstrate the μSP ATM during a four-month demonstration phase.

Code for AFAL selected missions will be developed and demonstrated as required. This task is intended as a level of effort software support phase. Estimated cost for Phase III and for Phase IV are shown in figure 77.

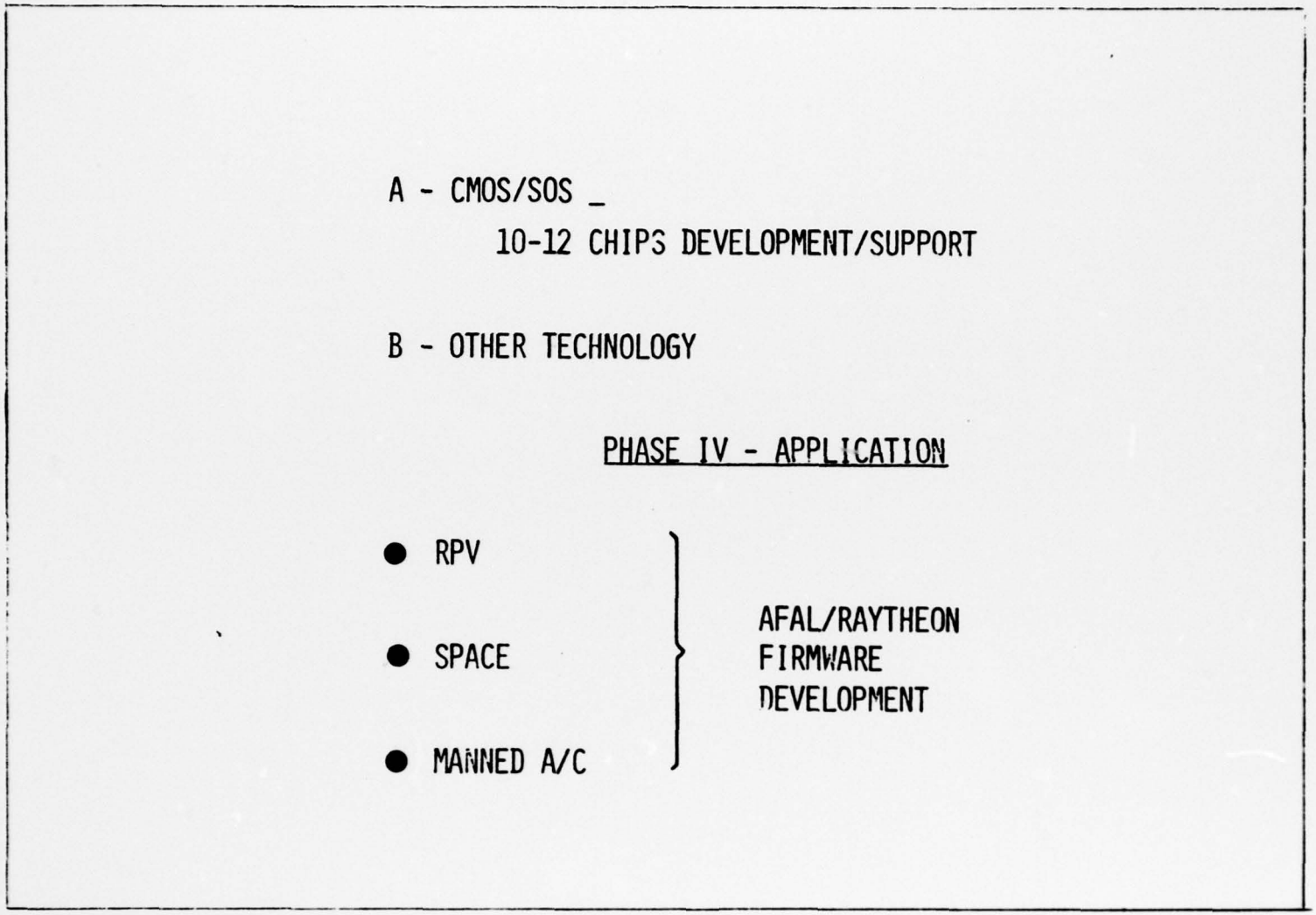


Figure 77 - Phase III - Advanced Technology Models

APPENDIX A
CONTRACT REQUIREMENTS

1. Contract Requirements

The microsignal processor program statement of work and Raytheon's fulfillment through schedule and cost are summarized herein.

1.1 Summary of Work Requirements

Tasks/Requirements

In achieving the objectives of this program the contractor shall accomplish the following tasks:

1.1.1 Functional Analysis: Functional requirements generic to airborne signal processing applications shall be defined.

Consideration shall be given to signal processing tasks inherent to radar, both air-to-air and air-to-ground modes including synthetic aperture ground map, electronic warfare, signal sorting and classification and communications, image/waveform coding and decoding. Algorithm classes to be considered and characterized shall include, but are not limited to:

- Digital Fourier transforms and inverse transforms to 2048 points.
- Digital filters, recursive and nonrecursive.
- Weighting functions, cosine squared, Taylor, Hanning, etc.
- Correlations, serial and parallel, various levels/combinations of source and reference signals.
- Walsh functions, Hadamard transforms, and related waveform/image coding transformations.
- Adaptive predictive coding, data bandwidth compression techniques.

- Nested polynomial functions, look up tables, etc, for common arithmetic computation.
- Integration, Averaging, and Standard Deviation
- Coodinate conversion

Processing tasks shall be tabulated with respect to typical performance parameters required of the signal processor in areas of:

- Word size - range and modularity
- Fixed, floating point arithmetic
- Operation mix - arithmetic/logic/control
- Processing rates - operations per second
- Input/Output - operations, rates, formats
- Memory - size, organization
- Data handling requirements
- Environmental/engineering constraints - size, weight, cooling, temperature, etc.

Performance ranges and levels, degrees of modularity and commonality of architecture/design features for a programmable signal processor shall be assessed.

1.2 Performance Analysis

From the data base developed under Task 1.1.1, a set of signal processing tasks (minimum of 4) shall be selected as a base reference for detailed performance requirements analysis. Insofar as practical, these tasks shall be representative of the complete spectrum of airborne signal processing requirements which are anticipated to be characteristic of platforms and avionic subsystems for the 1980-1985 era. Typical examples of application tasks include:

- Synthetic Aperture Ground Map
- Fast/Slow Moving Target Detection

- Terrain Following/Avoidance
- Ground Moving Target Track
- Air-to-Air Search and Track
- A/G, A/A Weapon Delivery
- RF Signal Sorting and Classification
- Image Compression/Jam Resistant Transmission

The baseline processing tasks shall be analyzed with respect to mathematical theory, task partitioning, and flow charting. Loading characteristics associated with subtasks of the computational decompositions and subtask interrelationships shall be described in detail. Critical subtasks shall be identified. Consistency of computational requirements across the tasks will be assessed. Candidate architectural features, hardware design considerations, and related software elements to structure a modularly expandable, programmable signal processor to accomplish the signal processing functions shall be identified.

Signal processing tasks to be considered under this Task shall be subject to prior approval of the Air Force project engineer. Candidate task listings shall be proposed by the contractor at an initial program review to be conducted at AFAL as soon as practical for work progress, but no later than 60 days after contract start.

1.3 State-of-the-Art Review

The industry LSI microprocessor technology base shall be reviewed. Existing state-of-the-art and projected developments over the next five (5) years shall be addressed for such elements as:

- Microprocessors
- Arithmetic Units
- Micro Program Controllers
- Programmable Logic Arrays

- Memories (RAM, ROM, PROM, EAROM)
- Interface Circuits
- Software/Firmware
- Design/Development Support Tools

The applicability and availability of these elements to signal processing tasks in general, and specifically to the baseline processing tasks of Task 1.2 shall be assessed for the development time frame identified in Task 1.7.

1.4 Functional Definition of Elements

Top level functional specifications shall be developed for a set of hardware micro-signal-processor circuit elements which form a basis for modularly configured, programmable signal processing capabilities indicated by the results of Tasks 1.1, 1.2. The analysis shall consider implementations necessary for operation and support of the micro-signal-processor elements. Functional descriptions shall address, but are not limited to, the following:

- Hardware Elements

element architecture (general)
performance (function, speed, word size)
internal data paths/flow, control points
interfaces (I/O, data, control)
physical/environmental characteristics

- Processor Configurations

processor architecture/modularity
I/O, control interfaces
memory (capacity, segmentation, organization)
interelement data-control flows
performance capability (thruput)

- Software

control word formats (macro/micro)
control algorithms and sequences
software elements
language
macro library
I/O software
sequence control
macro/micro simulator
macro/micro assembler

Major tradeoff rationales (cost, performance, reliability, maintainability, etc.) to support the hardware - software specifications shall be identified.

1.5 Simulation

The hardware - software elements/approaches identified in Task 1.4 shall be simulated to verify applicability to the baseline processing tasks of task 1.2. The simulation is to address the ability of the basic architecture and instruction repertoire to achieve the desired signal processing functions rather than on technology implementation. A discrete event simulator shall be developed and used for the demonstration-evaluation of the modularity and expandability features of the micro-signal-processing elements for the system environment. The simulator shall be written in Fortran IV.

1.6 Circuit Technology Review

Status and characteristics of integrated circuit technologies will be reviewed to determine their feasibility/suitability for LSI implementation of the elements defined in Task 1.4. Projected availability considerations will be addressed in addition to fundamental performance capabilities. Recommendations shall

be supported by tradeoff analysis.

1.7 Development Plan

A plan for the hardware and software implementation and performance demonstration to the micro-signal-processing elements shall be defined. The plan shall address task breakout, schedules and block funding requirements consistent with:

I. Requirements/Definition Phase: 7 months (this program)
II. Implementation Phase: 24 months
III. Performance Verification: 6 months

Potential advantages of accelerated and/or over lapped Phase II, III developments shall be assessed for the processor implementation approach developed under Tasks 1.3 - 1.6.

1.8 Program Reviews

The contractor shall conduct three program reviews. These reviews will be held at the AF Avionics Laboratory with review agendas to include briefings on all task progress to date and specific plans for the ensuing reporting period.

1.9 Cost & Schedule Performance

Raytheon's proposed schedule for the seven month program is shown in Figure 1.1; all tasks were completed on schedule, reports and presentations varied by a few weeks.

- Tasks 1 and 2, (Functional and Performance Analysis) overlap because functions requirements functions become an iterative loop in the proposed tops-down analysis.

- Task 3, state-of-the-art review, resulted in collection and collation of data to determine industry trends.
- Task 4, Functional Definition of Elements, was developed in parallel with the element definition.
- Task 5, Simulation. The simulator was developed in parallel with the element definition.
- Task 6, Circuit Technology Review was developed from literature search and used for Phase II and Phase III recommendations.

PROGRAM SCHEDULE - F33615-76-C-1339

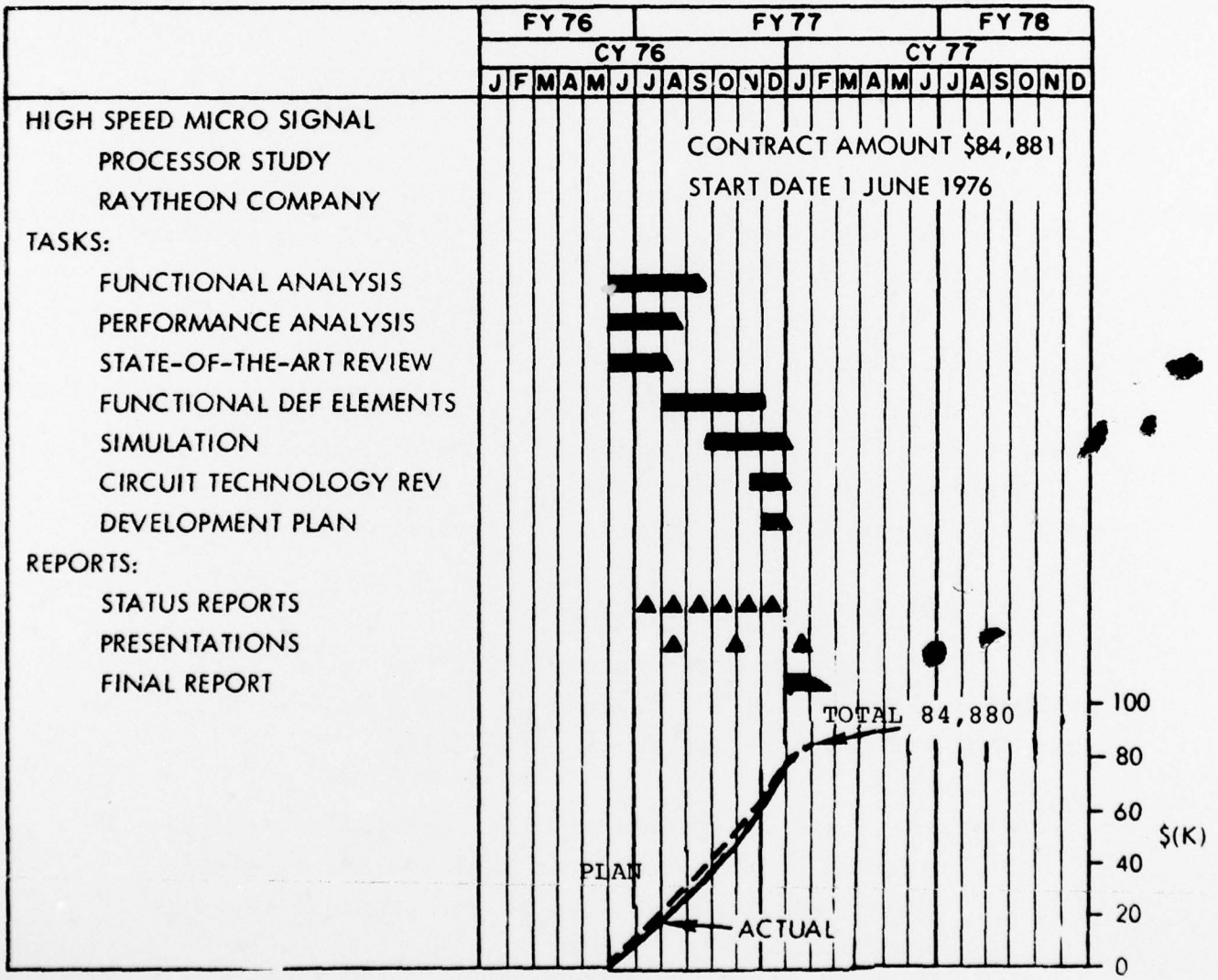


Figure 78 - Program Schedule

REFERENCES

1) John D. Markel and A.H. Gray Jr. "On Autocorrelation as Applied to Speech" IEEE Trans. Audio and Electro Acoustics Vol. AU-Z1, No 2 April 1973 pp 69-79.

2) N. Andersen, "On the Calculation of Filter Coefficients For Maximum Entropy Spectral Analysis," Geophysics, Vol. 39-No. 1 (Feb. 1974), pp 69-72.

3) Clifford J. Weinstein, "A Linear Prediction Vocoder with Voice Excitation" Eastcon '75 Record pp 30-A, to 30-G.

4) A.J. Goldberg and A. Arcese, "A Quantitative Comparison of Residual Encoding Techniques" Eastcon '75 Record, pp 28A to 28F.

5) J.D. Markel, "Digital Inverse Filtering a New Tool for Format Trajectory Estimation," IEE Trans. Audio Electroacout., Vol AU-20, pp 129-137, June 1972.

6) J.D. Markel, "The Sift Algorithm for Fundamental Frequency Estimation," IEE Trans, Audio Electroacoust., Vol AU-20, pp 367-377, Dec. 1972

7) J.P. Markel and A.H. Gray Jr., "On Autocorrelation Equations as Applied to Speech Analysis", IEEE Trans, Audio Electroacoust., Vol AU-21 pp 69-79, April 1973

8) A.H. Gray Jr. and John P. Markel, "Digital Lattice and Ladder Filter Synthesis," IEEE Trans. Audio Electroacoust., Vol AU-21, pp 491-500, December 1973.

9) J.D. Markel and A.H. Gray Jr., "A Linear Predictive Vocoder Simulation Based Upon the Autocorrelation Method," IEEE Trans, Acoust, Speech, Signal Processor, Vol ASSP-21, pp 124-134, April 1974

10) E.A. Robinson, Statistical Communications and Detection with Special Reference to Digital Data Processing of Radar and Seismic Signals, New York: AAFNER, 1967, pp 274-279.

11) Alfred Fettweis, "Pseudo Passivity, Sensitivity, and Stability of Wave Digital Filters," IEEE Trans, Circuit Theory, Vol CT-19, pp 668-673, November 1972.

12) Papers B5-10, Cr, D7, H10, and L4 in conf, rec, 1972 IEEE Conf. Speech Commun, and Processing.

13) N.E. Blackman, Sinasoids Versus Walsh Functions, Proc, IEEE Vol 62 pp 346-354, 1974.

14) D.S.K. Chan and L.R. Rabiner, IEEE Trans. Audio Electracoust., Vol. AU-21, pp 354-366, August 1973.

15) R.E. Crochiere and L.R. Rabiner, IEEE Trans, Acoust., Speech, Signal Processing Vol ASSP-23, pp 444-456, Oct, 1975.

16) R.E. Crochiere and A.V. Oppenheim, Analysis of Linear Digital Networks Proc, IEEE, Vol 63, pp 581-595, April 1975.

17) A.V. Oppenheim and R.W. Schafer, Digita. Signal Processing Prentice Hall 1975.

18) Eugene Hnatek, "Chipping Away At Core," Digital Design, July 1976, pp 31-42.

19) Jean Nicoud, "Peripheral Interface Standards For Microprocessors, Proc, IEEE, June 1976, pp 896-904.

20) A. Williams & H. Jelinek, "Introduction to LSI Microprocessor Developments," Computer, June 1976, pp 34-46.

21) EDN MICROCOMPUTER SYSTEMS DIRECTORY, Cahners Publishing Co., 1975.

22) MICROPROCESSOR SCORECARD, Microcomputer Techniques, Reston Virginia, Mini Micro Systems, July 1976.

23) MICROCOMPUTER D.A.T.A. BOOK EDITION 1, 1976, D.A.T.A. BOOK INFORMATION SERVICE, Orange, N.J.